Calhoun: The NPS Institutional Archive
DSpace Repository

Theses and Dissertations 1. Thesis and Dissertation Collection, all items

1975-12

Distributed microcomputer airborne tactical system.

Linhares, Tuxaua Plinio Barcelos de

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/20824

Copyright is reserved by the copyright owner

Downloaded from NPS Archive: Calhoun

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

DUDLEY KNOX LIBRARY

http://www.nps.edu/library

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

DISTRIBUTED MICROCOMPUTER AIRBORNE TACTICAL SYSTEM

Tuxaua Plinio Barcelos de Linhares

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA 93940

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DISTRIBUTED MICROCOMPUTER AIRBORNE TACTICAL SYSTEM

by

Tuxaua Plinio Barcelos de Linhares

December 1975

Thesis Advisor: U. R. Kodres

Approved for public release; distribution unlimited.

T170827

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (*and Subtitle*) Distributed Microcomputer Airborne Tactical System		5. TYPE OF REPORT & PERIOD COVERED Master's thesis; December 1975
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(*s*) Tuxaua Plinio Barcelos de Linhares		8. CONTRACT OR GRANT NUMBER(*s*)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE December 1975
		13. NUMBER OF PAGES 159
14. MONITORING AGENCY NAME & ADDRESS(*if different from Controlling Office*) Naval Postgraduate School Monterey, California 93940		15. SECURITY CLASS. (*of this report*) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (*of this Report*)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (*of the abstract entered in Block 20, if different from Report*)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (*Continue on reverse side if necessary and identify by block number*)

multiprocessor
airborne tactical system
microcomputer

20. ABSTRACT (*Continue on reverse side if necessary and identify by block number*)

An airborne tactical system composed of three distributed microcomputers is described. An extrapolation technique using the method of orthogonal polynomials is presented to solve the ballistics problem.

In order to test the performance of distributed microcomputers for real time military applications, a simplified airborne tactical system, utilizing two microcomputers working in parallel, is implemented. (cont.)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
(Page 1) S/N 0102-014-6601 |

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

20. (cont.)

The motivation for this research is the reduction of cost that would result from the use of microcomputers in such systems.

Distributed Microcomputer Airborne Tactical System

by

Tuxaua Plinio Barcelos de Linhares
Lieutenant Commander Brazilian Navy
B.S., I.M.E., Rio de Janeiro-Brazil, 1969

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL

December 1975

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA 93940

ABSTRACT

An airborne tactical system composed of three distributed microcomputers is described. An extrapolation technique using the method of orthogonal polynomials is presented to solve the ballistics problem.

In order to test the performance of distributed microcomputers for real time military applications, a simplified airborne tactical system, utilizing two microcomputers working in parallel, is implemented.

The motivation for this research is the reduction of cost that would result from the use of microcomputers in such systems.

TABLE OF CONTENTS

LIST OF FIGURES.................................... 6

I. INTRODUCTION.................................... 8

II. DESCRIPTION OF THE SYSTEM.................... 12

- A. THE NAVIGATION COMPUTER...................... 12
- B. THE BALLISTICS COMPUTER...................... 15
- C. THE EXECUTIVE COMPUTER....................... 15
- D. THE COMMUNICATION SCHEME..................... 16

III. THE BALLISTICS PROBLEM...................... 17

IV. FORTRAN SIMULATION........................... 32

V. THE MICROCOMPUTERS HARDWARE IMPLEMENTATION....... 40

VI. PL/M IMPLEMENTATION AND THE TEST CASES........... 43

VII. CONCLUSIONS.................................. 51

APPENDIX A:THE CONVERSION OF DATA PROGRAM AND OUTPUT.... 52

APPENDIX B:THE EXTRAPOLATING TESTING PROGRAM AND OUTPUT. 72

FORTRAN SIMULATION OUTPUT........................ 86

PL/M PROGRAM OUTPUT.............................. 98

FORTRAN SIMULATION PROGRAM....................... 106

PL/M PROGRAM..................................... 124

LIST OF REFERENCES............................... 158

INITIAL DISTRIBUTION LIST........................ 159

LIST OF FIGURES

1. System partial view.................................... 10
2. Visual description of the ballistics problem........ 18
3. The wind effect.. 20
4. Release point time determination...................... 38
5. Microcomputers hardware implementation.............. 42

ACKNOWLEDGEMENTS

The author wishes to express his thanks to those Instructors and Professors of the Computer Science Group, the Department of Mathematics and Department of Electrical Engineering of the Naval Postgraduate School, with whom he worked in a productive and pleasant environment.

Special thanks are due to Associate Professor Uno R. Kodres for the opportunity to engage in such an interesting project and for the permanent assistance during the whole thesis preparation.

The author is also grateful to Associate Professor Gary A. Kildall and Assistant Professor V. M. Powers for their encouragement.

Finally, many thanks to LCDR (Bras Nav) Luis R. B. Pedroso for the help received.

I. INTRODUCTION

Airborne tactical systems on board attack aircraft are designed to perform the following functions:

1. To navigate to a given target.

2. To control the auto-pilot.

3. To display information to the crew members.

4. To solve the ballistics equations that will compute the down range travel of the weapon.

5. To extrapolate from the available data, the release point of the weapon.

Current tactical systems employ a minicomputer as their main computing device.

An alternate approach to solve this problem is by the use of several microcomputers processing in parallel in place of the minicomputer. Although the most recent microcomputers are much faster than their predecessors, they still lag behind the minicomputers in their arithmetic processing capability. That is why a distributed system is needed. In such a system, the processors are put to work with as much parallelism as possible and they communicate with each other by means of an interrupt mechanism. The result is that the single computing cycle is broken up into smaller cycles some of which are executed in parallel, reducing the response time.

In this system, three microprocessors constitute an airborne tactical system:

1. The Executive computer.

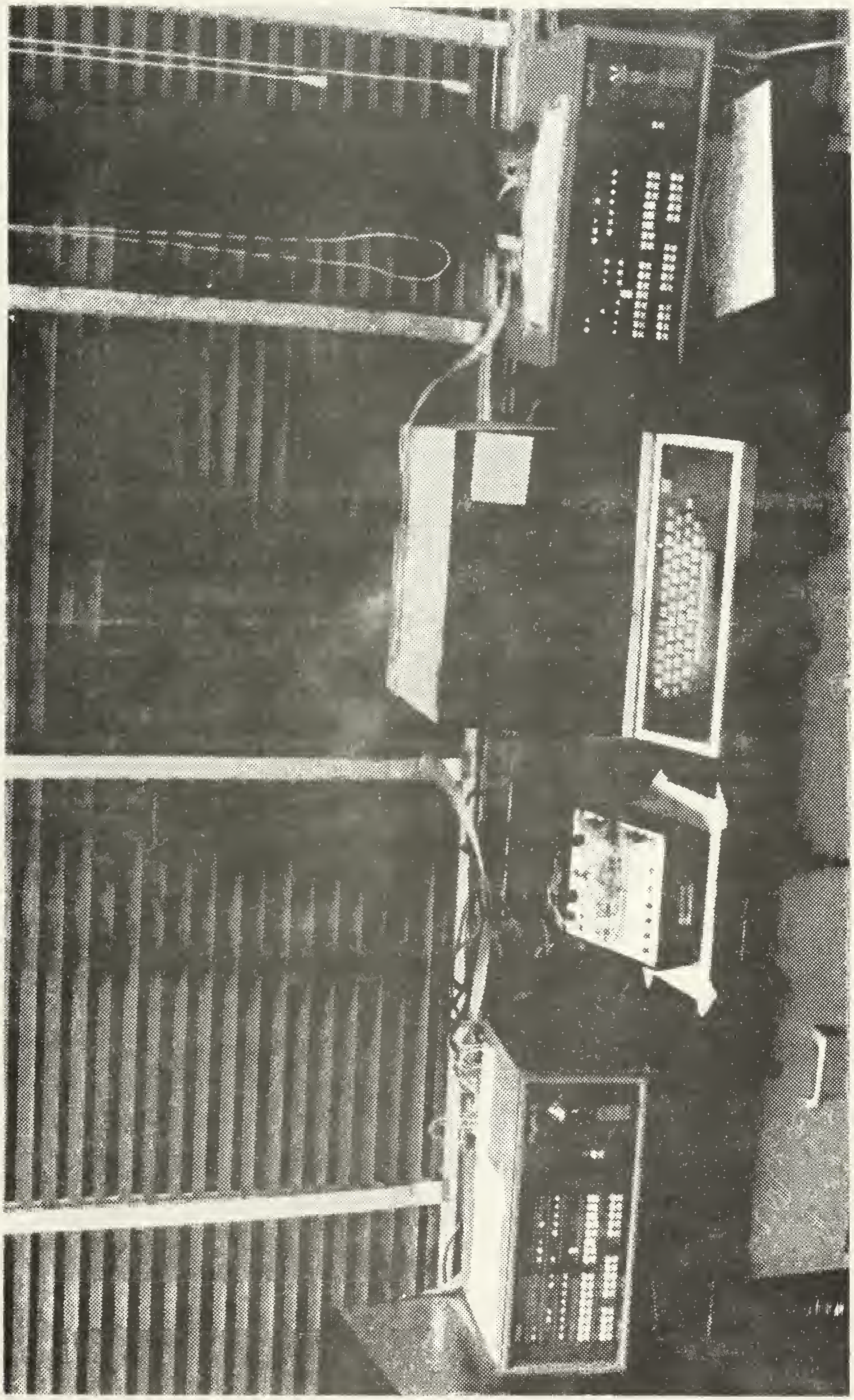
2. The Navigation computer.

3. The Ballistics computer.

Because of the availability of only two microcomputers, this thesis implements a simplified version of an airborne tactical system using two Intellec/8-Mod 80 microcomputers working in parallel to simulate the attack function of the system.

The system computes from a given series of aircraft positions with respect to the target the release point of the weapon so as to hit the target. All of this happens on a real time basis, controlled by an external clock.

Fig 1 shows a photograph of part of the system.



To provide the reader with an understanding of how a tactical system performs, Chapter II describes a system using three microcomputers.

Chapter III consists of an analysis of the ballistics problem which will be the critical part of the implemented system.

In Chapter IV, the work done on the I.B.M. System /360 with FORTRAN is described. The objectives of this phase of the project were mainly experimentation with system organization and design and testing of numerical methods.

Chapter V has a brief report about the necessary hardware adjustments and setup needed for the multiprocessing scheme.

Finally, in Chapter VI, the PL/M implementation is described, together with four test cases.

The data used were partially obtained from real runs performed at the China Lake target range.

Appendix A contains the FORTRAN program that was written to perform the conversion from the encoded data of range, elevation and azimuth of the aircraft, as tracked by a Nike-Zeus radar, into decimal representation of altitude and horizontal distance to target. The program also computes speed and dive angle, to be used to test the extrapolation routine. This same program, converts these values into the three byte floating point number format for the PL/M program.

Appendix B shows a program which tests the extrapolation routine.

II. DESCRIPTION OF THE SYSTEM

The system is composed of three computers, sensors, display units and auxiliary devices, each of which is described below. The reader may find a more complete description cf the sensor instruments in [2].

A. THE NAVIGATION COMPUTER

This computer executes a program that periodically polls sensors in order to compute the change in the aircraft's position during the last time increment.There are four sets of sensors to gather the necessary analogue data which is then converted to digital form and presented to the computer.These sensor instruments are:

1. The Inertial Navigation System (INS).

2. The Doppler Radar (DR).

3. The Air Data Computer (ADC).

4. The Radar Set (RS).

The INS is a gyro-stabilized platform with accelerometers mounted so that accelerations of the vehicle are measured in the north-south, east-west and vertical directions. It operates on the principle that every time the aircraft changes speed or direction it experiences an acceleration. With the continuous measurements of

aircraft's accelerations the INS determines the velocity by means of analogue integration.

A Doppler Radar radiates a pattern of beams to the surface of the earth and receives the reflection of this energy back. The difference between the frequency of the transmitted signal and the frequency of the received signal caused by the relative motion of the source of the signal with respect to the reflecting surface is known as the Doppler Effect. By measuring the frequency shift along each beam, the DR computes the velocity and distance travelled by the aircraft.

The Air Data Computer consists of the aerodynamics and thermodynamics sensors: angle of attack vane, static pressure source, Pitot tube and temperature probe. It computes the airspeed, the free-stream outside-air temperature and the Mach number.

In order to use the ADC for dead-reckoning, the attitude and heading of the aircraft must be supplied from other sources. This information together with the information of wind's velocity and direction, allows the determination of the aircraft's position by extrapolating from a previously known position.

The Radar Set determines the distance of the target by measuring the time elapsed from the transmission of a electromagnetic wave to its reception after being reflected by the target. The bearing is determined by the antenna's direction.

The system has a magnetic compass that indicates the aircraft's heading in relation to the magnetic north.

An altimeter supplies the system with altitude

information.

There are four basic modes of operation of this system: By using both the INS and the DR, the system operates in its most accurate mode. If one of these primary sources is malfunctioning, the system operates without its information, resulting in other modes of operation. In the case of failure of both INS and DR, the system operates in its least accurate mode using the ADC with the estimated wind vector and true airspeed of the aircraft to compute latitude and longitude.

No matter in which mode the system is operating, the information of aircraft's position is smoothed by the Navigation computer. The four most recently computed points are approximated by a quadratic polynomial using a least-squares fit. This smoothing process has a damping effect that reduces spurious oscillations of the data points. The mathematical description of this technique is found in Chapter III.

Using the smoothed data values, the increments in distance travelled in the last time increment are computed and the position coordinates displayed. As soon as the new values are computed, the Navigation computer interrupts the Executive and transfers this information to the Executive computer. The Navigation computer has the ability to function in a stand-alone mode to display aircraft location regardless of the status of the rest of the system.

Another important feature of the least squares curve fitting technique is the ease with which sensor malfunctions may be spotted. The crew is warned of the ill behavior of one of the sensors, and the system's mode of operation may be changed.

B. THE BALLISTICS COMPUTER

This computer receives from the Executive computer: aircraft speed, dive angle and altitude. The type of weapon and target's altitude are initially received from the Executive computer. With all these data, the computer numerically solves the differential equations to compute the down range travel of the weapon. There will be no guidance of the weapon after release, although the weapon may be rocket assisted or have a retarded mode of fall. When the Ballistics computer finishes its calculations, it interrupts the Executive computer to transfer the results.

C. THE EXECUTIVE COMPUTER

The Executive computer controls the function of the whole system. It sends to the Ballistics computer, at the beginning of the run, the target's altitude and the weapon selection. The Executive computer will be interrupted by the Navigation computer which periodically reports the aircraft's position. This information is sent to the Ballistics computer after it is converted into the proper form.

The number of data points that the Executive computer needs for the extrapolations depends on the degree of the fitting curve. In this system, it will be a third degree polynomial. In order to guarantee a solution to the extrapolation process, the Executive computer has to be supplied with at least four distinct data points.

When the Executive computer has enough data points representing horizontal distance to target and respective

down range travel together with the corresponding time, it extrapolates these two functions of time and finds their intersection. At the intersection of these two curves, the distance to the target will be the same as the down range travel. Thus, this will be the release point which is sought.

This computer will divide its time between data manipulation, data transmission, displaying of information, computation of release point and delivering steering commands to the auto pilot. If so desired, when the release point is reached, the computer will automatically trigger the release mechanism.

An external real time clock provides the Executive computer with exact time, so that time dependent variables may be recorded together with real time values.

D. THE COMMUNICATION SCHEME

Because of the inherent hierarchy between the computers involved, the master-slave type of multiprocessing is the most suitable and simple form to implement. The Ballistics and the Navigation computer act like peripheral devices of the Executive computer, resulting in a one way interrupting scheme. The only computer which has to be interrupted is the Executive computer. The Navigation and the Ballistics computers are the dedicated slaves which asynchronously interrupt the Executive computer.

III. THE BALLISTICS PROBLEM

In the most general form, this problem may be described as follows.

The aircraft's position in space is described at time t by a radius vector $\underline{r}(t)$ in a three dimensional coordinate system. The aircraft's velocity is described by the derivative with respect to time of the radius vector, i.e. the vector $d\underline{r}/dt=\dot{\underline{r}}$

These two vectors are shown in Fig 2.

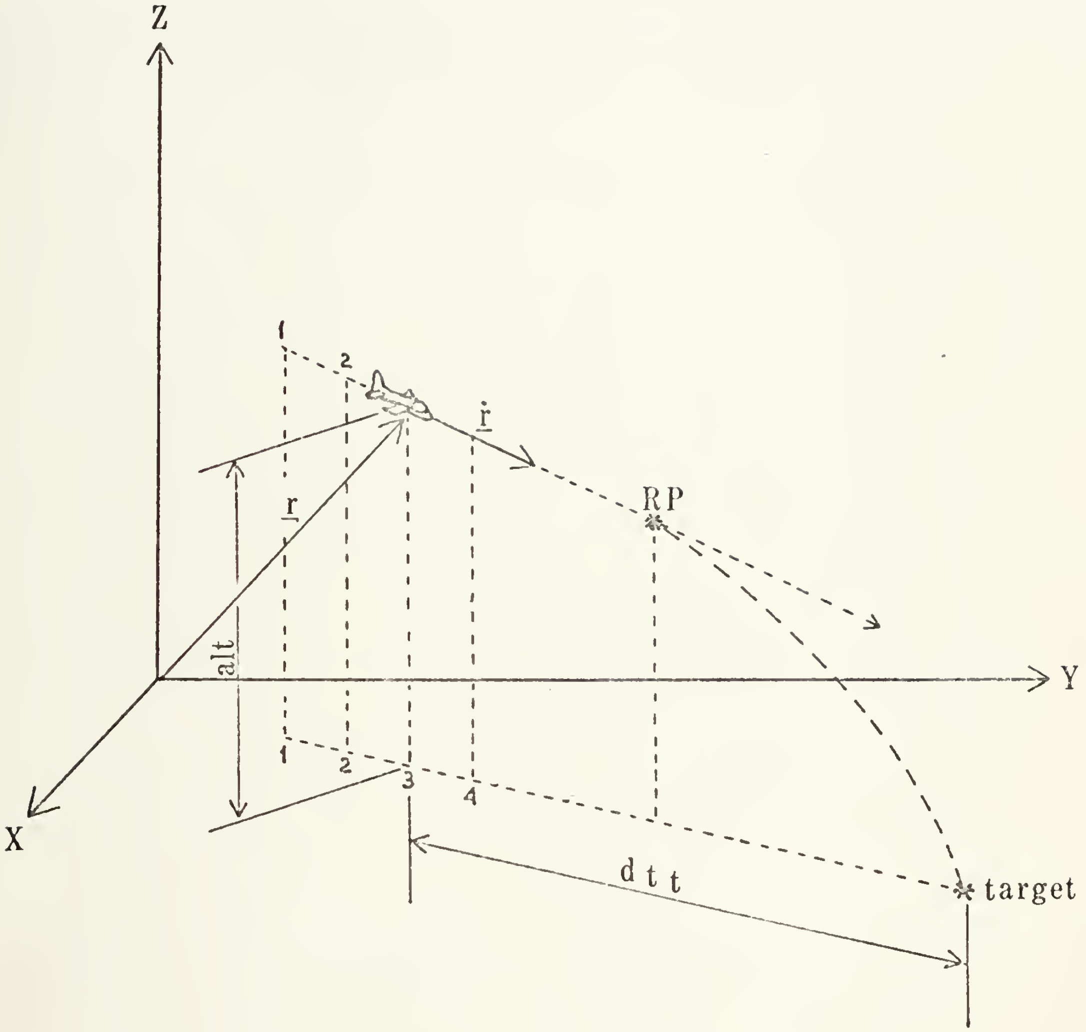


Figure 2 - VISUAL DESCRIPTION OF THE BALLISTICS PROBLEM

It is assumed throughout that the aircraft during its final approach to the target is restricted to fly in a vertical plane that contains the target and the origin, so that the problem becomes two dimensional.

In the presence of strong winds, the ground track of aircraft is used to determine the X-Y plane as Fig 3 shows.

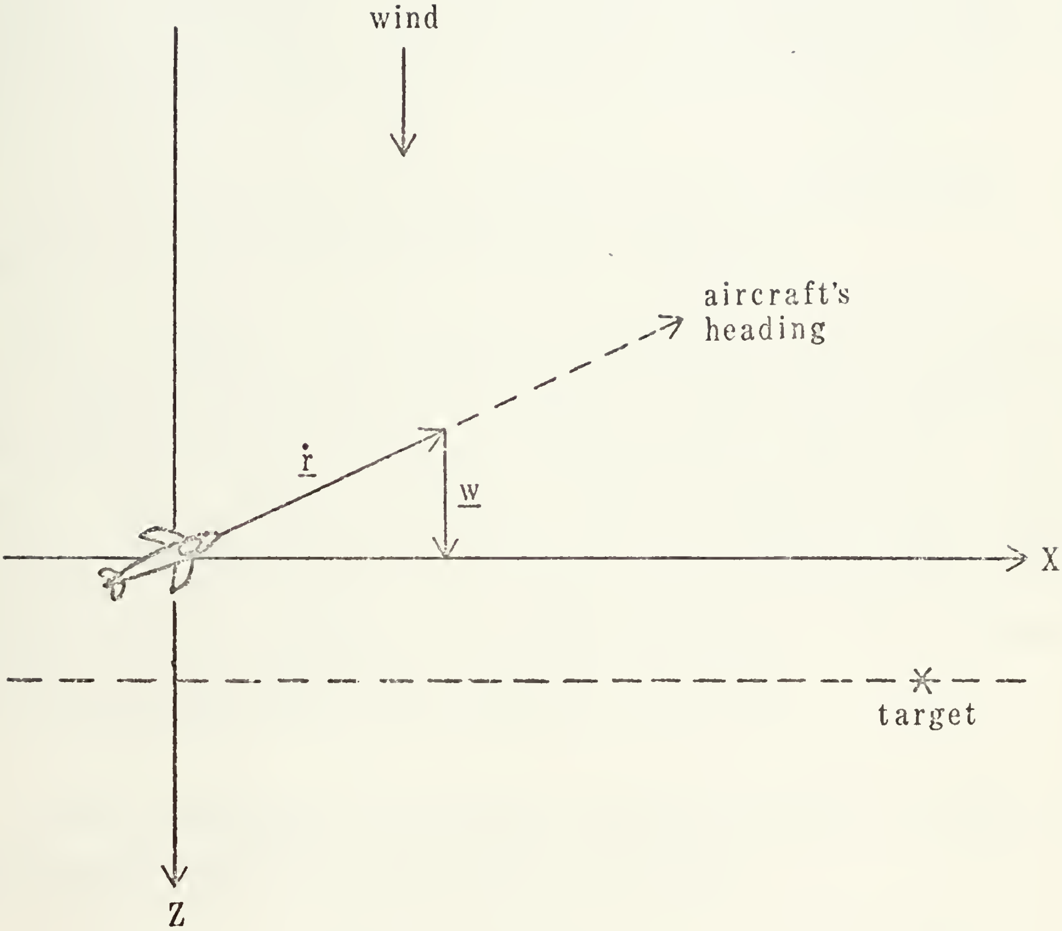


Figure 3 - THE WIND EFFECT

The target is located on a line which is parallel to the ground track of the aircraft, so that at the estimated release altitude the wind will carry the weapon through the distance between the ground track line and the dotted line parallel to the X-Y plane.

In the case that the wind direction coincides with the aircraft's heading, the target will be in the X-Y plane and the wind effect will either reduce or increase the down range travel of the weapon.

With the problem, now reduced to two dimensions, the following mathematical formulation applies:

At the instant the weapon is released from the aircraft, the initial position and velocity of the weapon agrees with the position and velocity of the aircraft. There may be a small added initial velocity when the weapons are released by small charges, or there may be a large added muzzle velocity for rockets and guns. In any case, the initial velocity of the weapon is determined by the aircraft's velocity $\underline{\dot{r}}(t_0)$ plus the constant vector $\underline{V}$.

After released, the weapon becomes a freely falling body satisfying Newton's differential equation

$$m\underline{\ddot{u}}=m\underline{g}-c\,|\underline{\dot{u}}|\,\underline{\dot{u}},$$

where $\underline{u}$ is the position vector of the weapon and the forces acting on the weapon are the gravity and drag which is proportional to the velocity squared. The term c is a characteristic drag term and it is weapon dependent.

In the two dimensional coordinate system, the equations

become $\underline{u}=(x,y)$.

$$m\ddot{x}=-c\sqrt{\dot{x}^2+\dot{y}^2}\,\dot{x}$$

$$m\ddot{y}=-mg-c\sqrt{\dot{x}^2+\dot{y}^2}\,\dot{y} \qquad (1)$$

with initial conditions:

$$x(t_0)=r_x(t_0) \qquad \dot{x}(t_0)=\dot{r}_x(t_0)+v_x$$

$$y(t_0)=r_y(t_0) \qquad \dot{y}(t_0)=\dot{r}_y(t_0)+v_y$$

The bombardier's problem is to determine the time t at which to release the weapon in order to hit the target. Stated mathematically, what needs to be solved is the combination initial value-boundary value problem:

$$m\ddot{x}=-c\sqrt{\dot{x}^2+\dot{y}^2}\,\dot{x} \qquad x(\bar{t})=r_x(\bar{t}) \qquad \dot{x}(\bar{t})=\dot{r}_x(\bar{t})+v_x$$

$$m\ddot{y}=-mg-c\sqrt{\dot{x}^2+\dot{y}^2}\,\dot{y} \qquad y(\bar{t})=r_y(\bar{t}) \qquad \dot{y}(\bar{t})=\dot{r}_y(\bar{t})+v_y$$

with the boundary condition that at time t^*:

$$y(t^*)=0 \qquad x(t^*)=T$$

In order to give a clearer understanding of the problem, let it be further simplified by assuming that the drag forces on the weapon are negligible. In this case, the differential equations become very simple. Let it also be that the release point $t_0=0$. The system becomes:

$$m\ddot{x}=0 \qquad x(0)=r_x(0) \qquad \dot{x}(0)=\dot{r}_x(0)+v_x$$

$$m\ddot{y}=-mg \qquad y(0)=r_y(0) \qquad \dot{y}(0)=\dot{r}_y(0)+v_y$$

The analytic solution to the initial value problem is obtained by integrating both equations twice with respect to t, resulting:

$$x(t)=(\dot{r}_x(0)+v_x)t+r_x(0)$$

$$y(t)=-gt^2/2+(\dot{r}_y(0)+v_y)t+r_y(0)$$

Next it is desired to find at what time t^* does the weapon hit the ground, i.e. at what time t^* is the condition $y(t^*)=0$ met.

This question is answered by solving the quadratic equation for the positive root:

$$-gt^2/2+(\dot{r}_y(0)+v_y)t+r_y(0)=0$$

giving:

$$t^*=(\dot{r}_y(0)+v_y)/g+\sqrt{(\dot{r}_y(0)+v_y)^2+2gv_y}\,/g$$

Therefore, in order that the weapon hit the target at time t^*:

$$T=x(t^*)=(\dot{r}_x(0)+v_x)t^*+r_x(0)\text{, or:}$$

$$T=(\dot{r}_x(0)+v_x)[(\dot{r}_y(0)+v_y)/g+\sqrt{(\dot{r}_y(0)+v_y)^2+2gr_y}\,/g]+r_x(0) \qquad (2)$$

As an example, if the aircraft is at level flight with constant speed:

$$\dot{r}_y=0 \qquad \dot{r}_x=\text{constant}=S$$

$$r_y=h, \text{ and}$$

$$T=(S+v_x)[v_y/g+\sqrt{v_y^2+2gh}\;/g]+r_x(0)$$

The release point $(r_x(0),h)$ is then given explicity by:

$$r_x(0)=T-(S+v_x)[v_y/g+\sqrt{v_y^2+2gh}\;/g]$$

Even in the general case, relation (2) allows the determination of the release point $(r_x(0),r_y(0))$ implicitly.

Therefore, if the drag effect is neglected, the initial-boundary value problem is relatively easy to solve. However, drag cannot be ignored in the real case, requiring that the more complicated system of differential equations (1) be solved.

If a dedicated computer is employed to solve the initial value problem (1) sufficiently fast, the down range travel $x(t_0^*)$ may be determined for any release instant t_0.

Therefore the release time $\bar{t}$ would be that value for which $x(\bar{t})=T$.

To solve this problem, all that is needed is an extrapolation technique, so that from successive aircraft

horizontal distances to target and corresponding computed down range travels it is possible to estimate future values for these two functions. At the release point these estimates should coincide.

The extrapolation technique employed is a least-squares approximation as follows.

It is desired to approximate a series of N data points $\{f(x_n)\}$ by a function of the form

$$F(x)=d_1u_1(x)+d_2u_2(x)+\ldots+d_ku_k(x),$$

that depends linearly on the parameters $d_1,\ldots,d_k$.

The functions $u_i(x)$, are selected a priori and the parameters must be determined.

In order to approximate the data points $f(x_n)$ by the function $F(x)$, the parameters must minimize the function

$$E(d_1,\ldots,d_k)=\sum_{n=1}^{N}[f(x_n)-F(x)]^2$$

This criterion function leads to the least-squares method.

By applying minimization techniques, one gets the following system of k linear equations in k unknowns:

$$d_1^* \sum_{n=1}^{N} u_i(x_n)u_1(x_n) + \dots + d_k^* \sum_{n=1}^{N} u_i(x_n)u_k(x_n) =$$

$$= \sum_{n=1}^{N} f(x_n)u_i(x_n) \qquad i=1,\dots,k \qquad (3)$$

It may happen that although this system always has a solution it turns out to be unreliable if the matrix A of the coefficients is ill-conditioned, i.e. if its condition number, $\mathrm{cond}(A)=|A|\,|A^{-1}|$ is too big.

Next it will be shown what the effects of using orthogonal polynomials are and how to generate these polynomials.

The cond(A) depends entirely on the choice of the functions $u_i(x)$. One way to garantee a small cond(A), is to use orthogonal functions, to get:

$$\sum_{n=1}^{N} u_i(x_n)u_j(x_n)=0 \qquad i\neq j$$

By such a choice of orthogonal functions the system (3) reduces to:

$$d_i^* \sum_{n=1}^{N} u_i(x_n)u_i(x_n) = \sum_{n=1}^{N} f(x_n)u_i(x_n) \qquad i=1,\dots,k$$

which, besides rendering reliable results, is much simpler to solve, reducing the computation involved.

The definition of the scalar product of two functions g and h that will be used is:

$$<g,h>=\sum_{n=1}^{N} g(x_n)h(x_n)$$

The functions are said to be orthogonal to each other if

$$<g,h>=0.$$

A sequence of polynomials $\{P_i(x)\}$ is said to be orthogonal if all $P_i(x)$ are orthogonal to each other and each $P_i(x)$ is a polynomial of exact degree i. One property of such a sequence is that any polynomial p(x) of degree $\leq k$ can be written as

$$p(x)=d_0P_0(x)+d_1P_1(x)+\ldots+d_kP_k(x) \qquad (4)$$

The measure of difference between f(x) and p(x) that needs to be minimized can be written as

$$<f(x)-p(x),f(x)-p(x)>=\sum_{n=1}^{N}[f(x_n)-p(x_n)]^2$$

And,

$$E(d_0,\ldots,d_k)=$$

$$=<f(x)-d_0P_0(x)-\ldots-d_kP_k(x),f(x)-d_0P_0(x)-\ldots-d_kP_k(x)>$$

The coefficients that produce the best fit have to satisfy the system of linear equations:

$$d_0^*<P_0,P_i>+d_1^*<P_1,P_i>+\ldots+d_k^*<P_k,P_i>=<f,P_i> \qquad i=0,\ldots,k$$

By orthogonality, this reduces to:

$$d_i^*\langle P_i,P_i\rangle=\langle f,P_i\rangle \quad i=0,\dots,k$$

Thus, if $S_i=\langle P_i,P_i\rangle\neq0 \quad i=0,\dots,k$

the coefficients will be given by:

$$d_i^*=\langle f,P_i\rangle/S_i \quad i=0,\dots,k$$

Now it will be shown how the sequence of orthogonal polynomials may be generated from the N data points.

These polynomials satisfy the three term recurrence relation:

$$P_{i+1}(x)=A_i(x-B_i)P_i(x)-C_iP_{i-1}(x) \quad i=0,1,\dots,k-1$$

$$B_i=\langle xP_i(x),P_i(x)\rangle/S_i \quad i=1,\dots,k-1$$

$$C_i=\text{arbitrary} \quad i=0$$

$$C_i=(A_iS_i)/(A_{i-1}S_{i-1}) \quad i>0$$

Where $A_i=a_{i+1}/a_i$ all i

$$P_{-1}(x)\equiv0$$

$S_i=\langle P_i,P_i\rangle\neq0 \quad i=0,\dots,k-1$ and a_i is the leading

coefficient of the polynomial P_i.

For simplicity it is decided to get all orthogonal polynomials with leading coefficient equal to 1, so that:

$$A_i = a_i = 1 \quad \text{all } i.$$

Setting $P_0(x) \equiv 1$,

$$S_0 = \langle P_0, P_0 \rangle = \sum_1^N 1 = N$$

Next, from the recurrence relations:

$P_1(x) = (x - B_0) P_0(x) = x - B_0$, where

$$B_0 = \langle xP_0(x), P_0(x) \rangle / S_0 = \sum_{n=1}^N x_n / S_0$$

With $P_0(x), \ldots, P_j(x)$ already obtained, the general step is:

$$S_j = \langle P_j, P_j \rangle = \sum_{n=1}^N [P_j(x_n)]^2$$

Because $P_j(x)$ is of exact degree j, S_j can be zero only if no more than j of the points $x_1, \ldots, x_N$ are distinct. If there are more than j distinct points among the data points, it is possible to calculate:

$$B_j = \langle xP_j(x), P_j(x) \rangle / S_j = \sum_{n=1}^N x_n [P_j(x_n)]^2 / S_j$$

$C_j = S_j / S_{j-1}$

The next orthogonal polynomial will be:

$$P_{j+1}(x) = (x - B_j) P_j(x) - C_j P_{j-1}(x)$$

As an example, the method will be applied to approximate the second degree polynomial

$$f(x) = 10 - 2x + x^2/10$$

at the points given by:

$x_n = 10 + (n-1)/5 \qquad f_n = f(x_n) \qquad n = 1, \ldots, 6$

$\sum_{n=1}^{6} [f_n - p(x_n)]^2$, needs to be minimized.

Thus,

$P_0(x) \equiv 1, \qquad S_0 = \sum_{1}^{6} 1 = 6$

and $B_0 = \sum_{n=1}^{6} [10 + (n-1)/5]/6 = 10.5$

$P_1(x) = x - 10.5 \qquad S_1 = \sum_{n=1}^{6} [(n-1)/5 - 0.5]^2 = 0.7$

$B_1 = \sum_{n=1}^{6} [10 + (n-1)/5][(n-1)/5 - 0.5]^2 / 0.7 = 10.5$

$C_1 = S_1 / S_0 = 0.1166667$

$$P_2(x)=(x-10.5)^2-0.1166667$$

Next the coefficients d_0^*, d_1^* and d_2^* for the least-squares approximation:

$p^*(x)=d_0^*P_0(x)+d_1^*P_1(x)+d_2^*P_2(x)$, are calculated:

$$d_0^*=\sum_{n=1}^{6} f_n/6=0.03666667$$

$$d_1^*=\sum_{n=1}^{6} f_n P_1(x_n)/0.7=0.1$$

$$d_2^*=\sum_{n=1}^{6} f_n P_2(x_n)/0.05973332=0.999999$$

Finally:

$$p^*(x)=0.03666667+0.1(x-10.5)+$$

$$+0.0999999[(x-10.5)^2-0.1166667]$$

IV. FORTRAN SIMULATION

It was realized that it would be advantageous to produce a FORTRAN simulation using the IBM System/360, because:

1. The computing environment has more power and the arithmetic is easier to handle than in PL/M.
2. The system's performance can be checked with respect to timing constraints.
3. The alternative computer program allows accuracy comparisons to be made with the PL/M implementation.

After the FORTRAN simulation had been tested and fully exercised, the PL/M program was then produced, taking advantage of the experience obtained during the development and use of the FORTRAN program.

In this simulation, the Navigation and the Ballistics computers are represented by subroutines and there has been no attempt to simulate the transfer of data mechanism between computers, because it was realized that the amount of effort required would not justify it.

The modules that comprise the simulation and their functions are:

MAIN	Governs the logic of the simulation and takes care of the real time clock.
INIT	Performs the initialization for each particular run.
ICHECK	Checks if there is a solution to the problem. If there is a solution, it then verifies if there is enough time to compute the solution. If the aircraft is too low and could be hit by the weapon's fragments, the pilot is warned.

DSPLAY	Displays the results and messages.
POS	Calculates the aircraft position and updates its airspeed.
BAL	Interfaces the MAIN module with the Ballistics modules.
SETDAT	Does the initialization process for the Ballistics computations.
DECODE	Sets parameters for the Ballistics computation, according to the selected weapon.
TRAJ	Calculates the downrange travel and time of flight of the selected weapon at the current aircraft position and speed.
RUNGE	Auxiliary module of TRAJ.
DERIV	Auxiliary module of TRAJ.
INTER	Commands the intersection search pattern for the horizontal distance to target and the downrange travel curves.
COMP	Does a stepwise comparison between the horizontal distance to target and the downrange travel curves, searching for their intersection.
ORTFIT	Extrapolates values for the horizontal distance to target and the downrange travel curves for a given time.

PROGRAM VARIABLES:

VARIABLE	DEFINITION
A	Runge Kutta parameter.
AA	0.5/Runge Kutta parameter.
ALT	Weapon release altitude.
AN1,AN2	Runge Kutta variables (X direction)
AP1,AP2	Runge Kutta variables (Y direction)
CC	Matrix of drag curve coefficients.
CF	CKDG stretch factor.
CFORM1	CKDG stretch factor for the first stage.
CFORM2	CKDG stretch factor for the second stage.
CKDG	Bomb coefficient times drag coefficient.
CM	Mach number.
CT	Matrix of Mach curve coefficients.
DEG	Weapon release angle.
DEL	ArcTan (ejection velocity/weapon velocity)
DKG	Shift in CKDG
DKG1	Shift in CKDG for the first stage.
DKG2	Shift in CKDG for the second stage.
DM	Shift in Mach number
DMAX	Largest integration step size.
DM1	Shift in first stage Mach number
DM2	Shift in second stage Mach number.
DS	Integration step size.
DTI	Initial integration step size.
FN	Equal to the thrust for the first stage.
FRACT	Parameter for Runge Kutta integration
G	Acceleration due to gravity.
HH	Total drag function.
IBOTH	Key determining if two drag curves are required by the weapon.
IDNO	Identification number of each weapon.
IREF	Reference for Mach curve coefficients.
IREG	Mach region index.
MSTG	Weapon stage index.

RAD	Conversion factor for radians.
RHO	Air density.
SL	Slope coefficient for retarded snakeye fins deployment time.
T	Elapsed time of flight from weapon release.
TH	Thrust/mass of rocket.
THETA	Adjusted weapon release angle.
U	Aircraft speed in feetSeconds.
V	Velocity of weapon.
VX	Component of V in the X direction.
VXA	VX at release point.
VXO	VX at start of present integration step
VY	Component of V in the Y direction.
VYA	VY at weapon release.
VYO	VY at start of present integration step
VZ	Muzzle velocity of guns.
X	Weapon ground range from release.
Y	Weapon altitude above sea level.
YO	Y at start of present integration step.
YT	Target altitude above sea level.
VKTS	Speed of aricraft in knots.
HV	Horizontal component of VKTS.
VV	Vertical component of VKTS.
HVA	Previous value of HV.
VVA	Previous value of VV.
XAT	Horizontal distance to target.
TIME	Real time.
FTIME	Saved value of TIME for DSPLAY.
HTIME	Previous value of TIME.
DTBAL	Time spent by the Ballistics Computer.
DTE1	Time spent by the Executive Computer to transfer data from the Navigation Computer.
DTE2	Time spent by the Executive Computer to transfer data from the Ballistics Computer.
DTE3	Time spent by the Executive computer to

	compute the release point.
DTE4	Time spent by the Executive Computer to execute check number two.
DTE5	Time spent by the Executive Computer to transfer data from the Navigation Computer and execute check number four.
DTE6	Time spent by the Executive Computer to save and restore values when calling the Navigation Computer and execute check number three.
TC	Time necessary to compute for the first time the release point.
TARP(2)	Time at release point.
XARP(2)	Horizontal distance to target at release point.
AARP	Aircraft altitude at release point.
TMV	Time it takes for the aircraft to start the evasion.
ATOL	Limit angle between aircraft path and line of sight of the target.
RLIM	Radius of a hemisphere defining a danger zone around the target.
ALIM	Limiting altitude for the aircraft.

The units of the variables are:

Distances: Nautical miles.
Altitudes: Feet.
Angles: Degrees.
Time: Seconds.
Speed: Knots.

The values of DTBAL, DTE1, DTE2, DTE3, DTE4, DTE5, DTE6, TMV, ATOL, RLIM and ALIM were assumed in this simulation. Most of them are weapon dependent.

This program used a least-squares fit of the third

degree, by means of orthogonal polynomials [3]. After the fourth aircraft position and corresponding downrange travels are computed, the Executive Computer proceeds to compute the first release point data. From this time on, for every new position and corresponding downrange travel received, the Executive Computer determines an updated release point using the four most recent aircraft positions with their downrange travels. This process goes on until there is no more time to compute the next release point.

The way the release point is found is by a stepwise comparison between the extrapolated values of the horizontal distance to target curve with the downrange travel curve. At the release point, these two curves intersect. After the curves have been extrapolated from the first four points, the comparisons between the extrapolated values of the two functions are made from point number five on. When the intersection point is passed, the direction of the search is reversed and the time increment is reduced. The search continues until the difference between the curves is smaller than a fixed limiting value.

Fig 4 shows graphically how the search for the intersection is made.

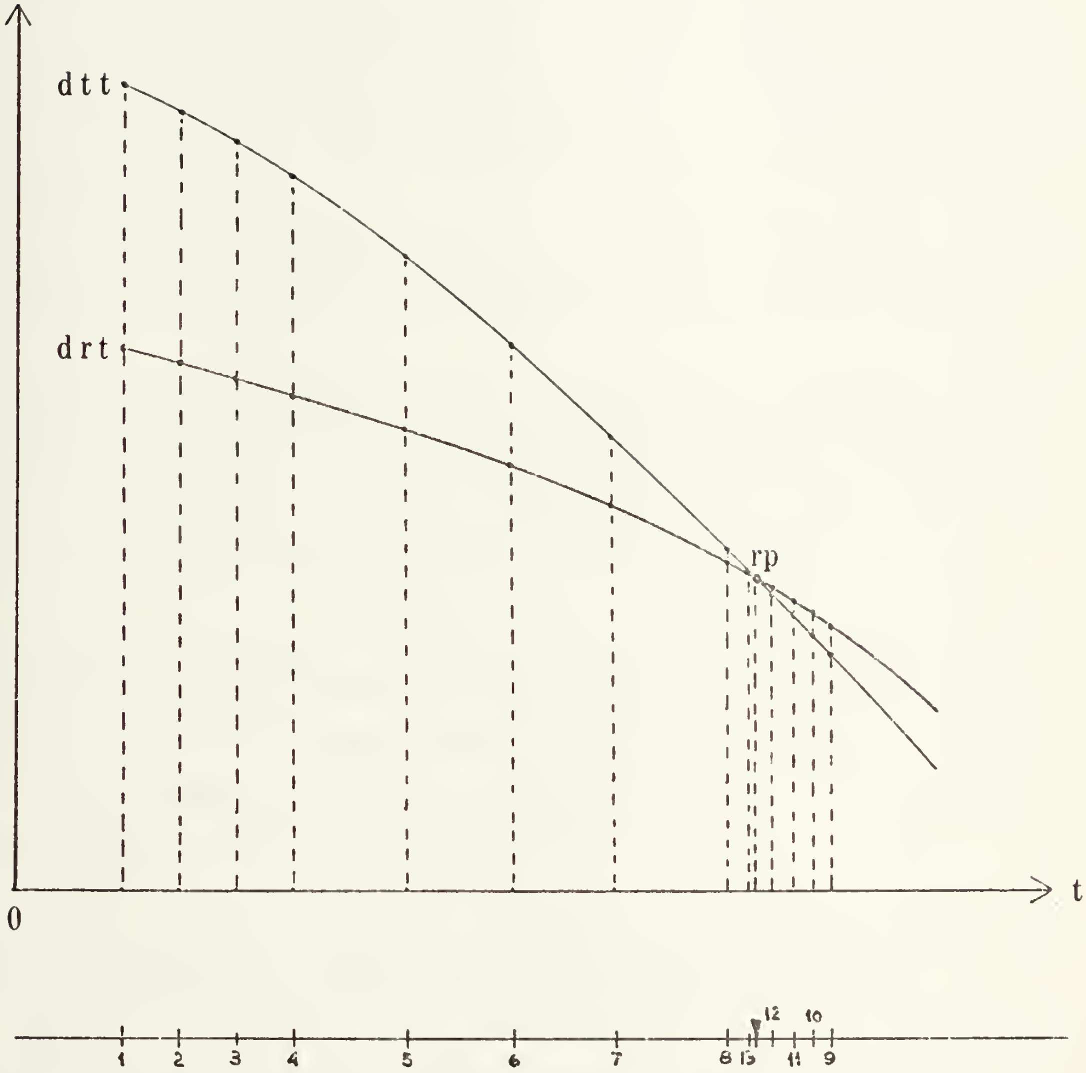


Figure 4 - RELEASE POINT TIME DETERMINATION

The ICHECK module has 4 entry points and its function is to check for exceptions. First, with the initial data, it checks if there is a solution to the problem. There are two situations that would not have a solution:

1. The aircraft dive angle is such that the impact of the weapon would always be short. This happens if, for a given weapon, the angle between the aircraft path and the line of sight of the target is less than a certain threshold.

2. The aircraft is not diving and the present distance to target is smaller than the downrange travel.

There is another situation in which the solution should not be computed, because at the release point the aircraft would be below its limiting altitude. In this case the system would also display "NO SOLUTION TO THESE DATA".

Because the computation of the release point requires approximately one second of real time, it is necessary first to check if the extrapolation can be carried out in time to release the weapon. If this can not be done, the system will display "NOT ENOUGH TIME TO COMPUTE".

Every time a release point is computed, the ICHECK module checks if at the moment of weapon impact the aircraft is inside a danger zone which is defined by a hemisphere around the target whose radius depends mainly on the weapon used. If this is the case,the system will display:"AIRCRAFT IS IN DANGER ZONE".

After each release point calculation the ICHECK module verifies if there is time to compute a new release point for the next position.

V. THE MICROCOMPUTERS HARDWARE IMPLEMENTATION

The hardware implementation of the system required that the two computers have a second input-output board installed, in order to provide all the connections needed.

To make the system simple, it was devised so as to use the fewest number of input-output ports possible. One of the key facts that allowed reduction of the communications paths among the computers was the predictability of message lengths throughout the complete cycle of system operation. The result is that there is no necessity of a status port during transmissions of data.

The Intellec microcomputer, as it is furnished by the manufacturer, does not have available externally the eight lines that constitute the interrupt port, nor does it have an external interrupt request line. It was necessary to set up a board to be installed on the Executive computer, to bring out these lines, so that the necessary connections could be made. Fortunately, this turned out to be the only change necessary to use the interrupt feature.

Another addition to the Executive computer was the external clock with an eight bit binary counter that was connected to one of its input ports.

Fig 5 shows the system's components. In this figure, the numbers correspond to the eight bit input-output ports of the computers. PTR stands for paper tape reader and DU stands for disk unit. In the Executive computer, the Interrupt Instruction port (II) and the Interrupt Request

line (Int) are shown. The Executive computer will be interrupted by means of the Restart 3 instruction, which has the bit pattern DF.

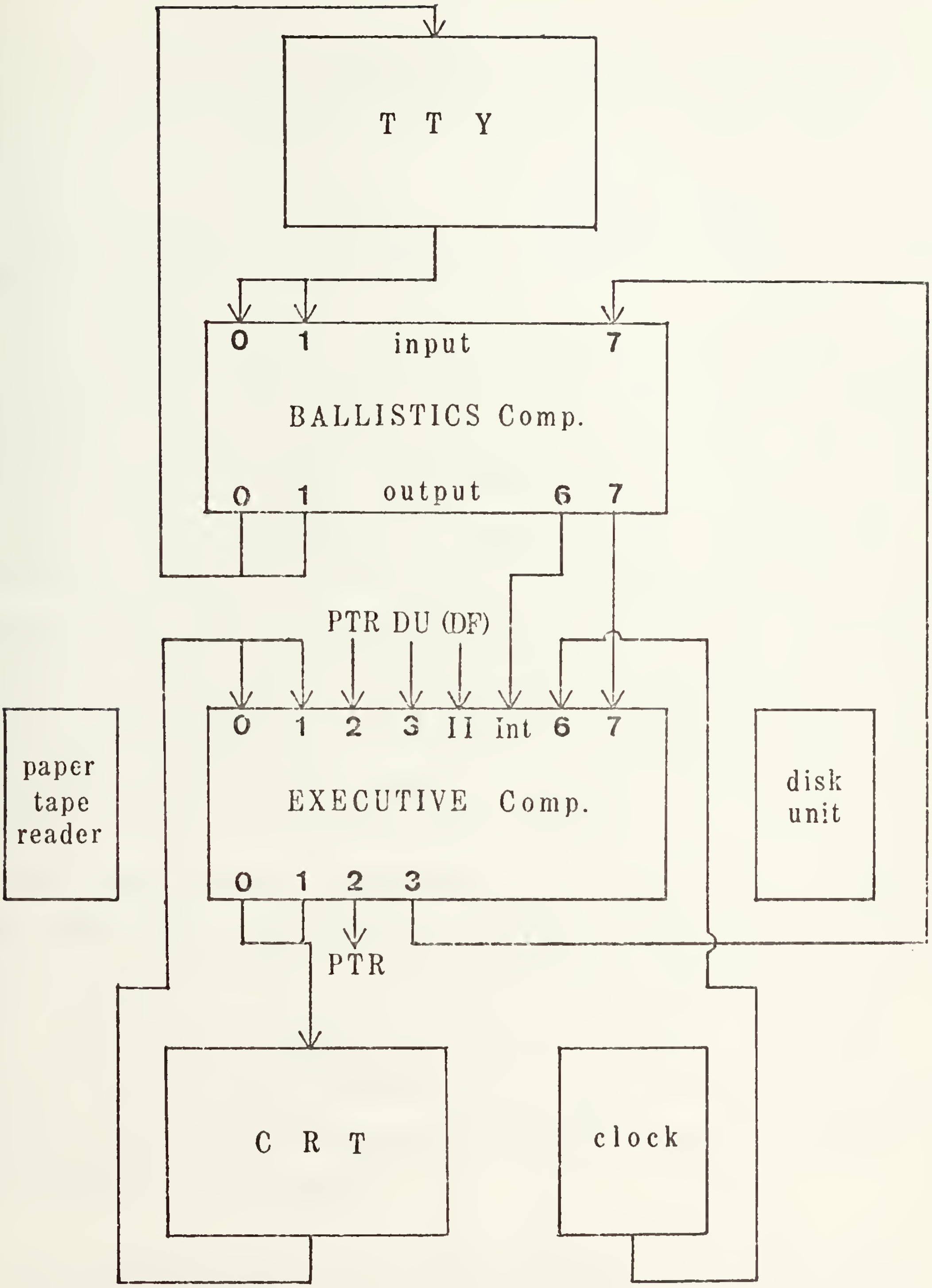


Figure 5 - MICROCOMPUTERS HARDWARE IMPLEMENTATION

VI. PL/M IMPLEMENTATION AND THE TEST CASES

The system's software is composed of two separate programs, one for each computer.

The program that runs on the Ballistics computer has two procedures to take care of the communication with the Executive computer. The first one is to receive the weapon identification number and target altitude at the beginning of the run. The other is to receive speed, dive angle and altitude of the aircraft and, subsequently, compute the down range travel. The module that computes the down range travel was the subject of a previous thesis [1] and is used in this system with only minor adaptations. When the computation of the down range travel is finished, the Ballistics computer interrupts the main computer in order to send back the results. If there is already data for the next computation of down range travel, this same procedure will receive the new data and proceed to compute. Otherwise the Ballistics computer will exit from this procedure and return to the main program. In this fashion the system guarantees that the Ballistics computer will be used at its maximum. This policy had to be adopted in view of the fact that the computation done in the Ballistics computer is the most time consuming in the system.

The system works in such a way that the Executive computer will only send data directly to the Ballistics computer if it is idle, in which case the Ballistics computer will be waiting for data, no interruption needed. If the Executive computer has data to send but the other computer is busy, a condition that the Executive computer

knows by testing a flag, the data is sent to a buffer.

The interrupt service procedure in the Executive computer is the one that upon completion of receiving the computed value from the Ballistics computer, checks if the buffer has data yet to be processed by the Ballistics computer. If this is the case, it will take care of transmitting it to the other computer.

The program that runs in the Executive computer is composed of the following modules, with a brief description of their functions:

Mon1: Links the program to the I/O facilities of the CP/M system if no returned value is expected.
Mon2: The same as above, except that a returned value is expected.
Read: Reads characters from the CRT into a buffer.
Print: Displays the buffer starting at the specified address.
Printchar: Displays the specified ASCII character.
Crlf: Sends carriage-return-line-feed characters.
Getc: Points to the current character in the buffer.
Readc: Reads one character from the buffer.
Readp: Reads a pair of characters from the buffer and returns the corresponding decimal value.

All of the above modules are needed to use the CRT as an I/O device by the CP/M system [4].

Conv: To return the decimal value of the number character.
Trans: Returns the ASCII character.
Clock: Reads in the values of the external clock.

Realt:	Displays real time.
Relpt:	Computes and displays the release point.
Sendbuff:	To send data to the buffer.
Dsplv:	Displays the variable value.
Display:	Displays aircraft's speed, dive angle and altitude.
Polin:	Computes the coefficients and constants used by the fitting plynomial.
Ftab:	Floats the byte values of time.
Search:	Finds the intersection of the curves being extrapolated.
Round:	Chooses the closest time to the intersection point, in decimals of second.
Extrap:	Given the time,extrapolates the value of the function.
Negat:	Tests if the variable is negative.
Mant:	Converts the mantissa of the number to decimal value.
Poten:	Computes the decimal value of the exponent.
Integ:	Displays the integer part of the values.
Decim:	Displays the decimal point and one decimal place fraction of the variable.
Index:	Finds the location of data in NAVTAB given the time.
Comp:	Auxiliary module of Search.
Intserv:	Interrupt service procedure. Receives computed values from the Ballistics computer. If the buffer is full, it sends new data to the other computer.

The simulation of the Navigation computer is accomplished by the use of a table with values of speed, dive angle, altitude and horizontal distance to target at points in time one second apart for the first ten points and

one tenth of a second apart for the last thirty points.

The first part of the table is used to obtain points to extrapolate values of horizontal distance to target. From the calculated values of down range travel using these same points, extrapolations of this variable are made.

The second part of the table has finer time increments in order to allow a more precise determination of aircraft's position at release point.

The complete cycle of operation of the system is as follows:

1. After checking if the Ballistics computer is ready, the Executive computer waits for inputs of run identification number and weapon identification number by CRT.
2. The Executive computer sends the weapon identification number to the Ballistics computer.
3. The Executive computer waits until the clock reads zero seconds.
4. The Executive computer brings from the NAVTAB table to a working area, the data corresponding to the present time.
5. These data are sent to the Ballistics computer if it is idle or to a buffer if it is busy.
6. The Executive computer displays the data sent, on the CRT.
7. The Executive computer waits until two seconds after the last point was sent to the Ballistics computer.

Steps 4, 5 and 6 are executed three more times.

During the time these events are taking place, the Ballistics computer calculates the down range travel and interrupts the Executive computer in order to send the

results of its computations.

There will be two instances where the Executive computer interrupts will be disabled: during buffer accesses and during displaying of data on the CRT.

8. When the Executive computer already has the down range travels for all four points, it computes the release point.

9. With the computed time for the release of the weapon, the NAVTAB table is accessed in order to obtain the corresponding position of the aircraft.

10. This data is sent to the Ballistics computer.

11. The Ballistics computer calculates the down range travel and sends it back to the Executive computer.

12. This result is displayed on the CRT.

The system returns to step 1.

The two second interval between the data points used for the extrapolation was adopted because of the displaying of information on the CRT during the time the computers are computing the release point. This display function takes on the order of one second to be performed. In a real implementation, the displaying would be avoided during the release point computation, because it would slow down the process.

This program was exercised with four sets of data. Runs one and two used real data that was obtained from a digital recording of radar tracking during A7 aircraft practice bombing runs at China Lake. Run three used data generated to represent level, constant-speed flight. Run four used constant dive angle and speed.

The sequence of displayed messages on the CRT, for runs one and three, were photographed. These photographs appear

in the section entitled PL/M program output, starting on page 98.

In the following table, the Fortran and the PL/M results are summarized.

The PL/M results are the real internal values, not the displayed values. The Display routine gives only approximate values.

This table shows for each run, the computed values of time of weapon release and corresponding aircraft's distance to target and down range travel. It also shows the actual values of distance to target and down range travel at the computed release time.

FORTRAN AND PL/M RESULTS:

		Fortran	PL/M
Run #1	Release point:		
	Time	10.84	10.7
	Distance to target	1831.4	1865.6
	Down range travel	1831.4	1870.5
	Tabulated values:		
	Time	10.84	10.7
	Distance to target	1831.5	1864.3
COMPUTED DOWN RANGE TRAVEL		1852.5	1888.0
Run #2	Release point:		
	Time	11.14	11.5
	Distance to target	1895.2	1817.0
	Down range travel	1895.2	1820.6
	Tabulated values:		
	Time	11.14	11.5
	Distance to target	1894.7	1817.5
COMPUTED DOWN RANGE TRAVEL		1891.7	1818.7
Run #3	Release point:		
	Time	11.18	11.0
	Distance to target	1837.3	1863.6
	Down range travel	1837.3	1861.3
	Tabulated values:		
	Time	11.18	11.0
	Distance to target	1836.0	1866.0
COMPUTED DOWN RANGE TRAVEL		1837.2	1861.0
Run #4	Release point:		
	Time	11.26	11.3
	Distance to target	2456.6	2444.2
	Down range travel	2456.6	2434.6
	Tabulated values:		
	Time	11.26	11.3
	Distance to target	2456.0	2446.1
COMPUTED DOWN RANGE TRAVEL		2455.9	2486.4

The results show two kinds of disagreements: between FORTRAN and PL/M results and also between the extrapolated and computed values of down range travel.

Both disagreements are due to:

1. The PL/M program that runs in the Ballistics computer to calculate the down range travel of the weapons is producing results that differ from its FORTRAN counterpart. This fact was not observed before. It so happens that with the data used in this implementation the PL/M program was not as accurate as it had been in previous tests.

2. The extrapolation of down range travel uses computed values, which in spite of being originated from smoothed data, contain roundoff errors. When a third degree polynomial is fit to four of these points, the noise present disturbs the extrapolation. A second degree polynomial through the same four points would probably give better results.

The time the Executive computer took to find the release point was on the order of 0.9 seconds, which is somewhat long. This time can be reduced by using a faster extrapolation technique.

VII. CONCLUSIONS

1. The hardware system used to demonstrate the tactical system is complete and working properly.

2. A different extrapolation technique is necessary in order to produce faster and more accurate results for the down range travel.

3. The Ballistics computer is time critical in this system and because of that the Executive computer should not hold up the Ballistics computer.

This thesis was not able to prove that the implemented system would meet the requirements of a real tactical system. It merely provides a framework with which further tests could be exercised. These tests should include different methods of extrapolations and all types of weapons. In particular, with respect to the extrapolation technique, a straight line fit through the last three data points seems to be promising both in accuracy and speed of computation.

APPENDIX A

THE CONVERSION OF DATA PROGRAM AND OUTPUT

This appendix shows the program that performed the conversion from the encoded values of range, elevation and azimuth of aircraft as tracked by radar into decimal representation used by the program to test the extrapolation routine and also to produce the data for the PL/M program which uses a three byte floating point number representation.

Following the program listing, the outputs are shown.

The floating point number used is composed of three bytes, as follows: The two higher order bytes constitute the mantissa. The first bit of the mantissa is always one, except in the case of the number zero, i.e. the number is a normalized binary number.

The lower order byte represents the exponent in the following manner: The first bit is the mantissa sign bit. If the number is negative, this bit is one. The following bits represent the exponent of two. This exponent is biased by hexadecimal forty, i.e. from this number and up, the exponent is positive and from this number down, the exponent is negative.

To make it clearer, some examples are given:

PL/M floating point number	decimal value
E1 00 49	450.0
A0 00 C4	-35.0
80 00 3F	0.25

```
C     ************************************************************
C     *                                                          *
C     *    PROGRAM TO PRODUCE A CARD DECK FOR THE DATA           *
C     *    CORRESPONDING TO AIRCRAFT POSITION,SPEED AND          *
C     *    DIVE ANGLE FOR EACH RUN.                              *
C     *    EACH RUN WILL HAVE 80 CARDS AND THE CARDS WILL        *
C     *    CONTAIN 16 TWO-HEXADECIMAL-CHARACTERS BYTES,          *
C     *    REPRESENTING: TIME, SPEED, DIVE ANGLE,                *
C     *    ALTITUDE AND DISTANCE TO TARGET.                      *
C     *    THE FIRST FOUR RUNS WERE OBTAINED FROM REAL           *
C     *    DATA AND THE LAST TWO RUNS HAVE BEEN                  *
C     *    GENERATED TO REPRESENT IDEAL CASES.                   *
C     *                                                          *
C     ************************************************************
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
      COMMON/ALL/ IARRI(16),IARRE(8) ,IARRF(16),KI,INC
      COMMON/MAKH/IZERO,ITWO,ITHREE,IFOUR,IFIVE,ISIX,ISEVEN,
     1IEIGHT,ININE,ITEN,IELEVN,ITWELV,ITHIRT,IFOURT,IFIFT,
     1IONE
      COMMON/MASH/ IHA(26)
      COMMON/MASD/SALT,SXAT,FALT,FXAT
      COMMON/MAORT/ ALTSM(40), XATSM(40), TIME(40),
     1              ALA(40), XAT(40),DRT(40)
      DIMENSION VELOC(40),DIVE(40)
      DIMENSION IEL(40), IAZ(40), IRAN(40), ITIME(40),
     1          ELEV(40), AZ(40), RAN(40)
    7 FORMAT(A1)
      READ(5,7)IZERO
      READ(5,7)IONE
      READ(5,7)ITWO
      READ(5,7)ITHREE
      READ(5,7)IFOUR
      READ(5,7)IFIVE
      READ(5,7)ISIX
      READ(5,7)ISEVEN
      READ(5,7)IEIGHT
      READ(5,7)ININE
      READ(5,7)ITEN
      READ(5,7)IELEVN
      READ(5,7)ITWELV
      READ(5,7)ITHIRT
      READ(5,7)IFOURT
      READ(5,7)IFIFT
      DO 101 IRUN=1,4
      WRITE(6,9)
      WRITE(6,60)
   60 FORMAT(/////)
    9 FORMAT('1')
      WRITE(6,1) IRUN
    1 FORMAT(11X,' RUN ',I1,':')
      WRITE(6,10)
   10 FORMAT(/)
      IF ((IRUN .EQ. 3) .OR. (IRUN .EQ. 4)) GO TO 41
      WRITE(6,70)
   70 FORMAT(12X,'INPUT DATA: CODED OCTAL DIGITS',6X,
     1'CONVERSION TO DECIMAL:')
      WRITE(6,71)
   71 FORMAT(12X,'ELEVATION',3X,'AZIMUTH',6X,'RANGE',6X,
     1'ELEVATION',1X,'AZIMUTH',1X,'RANGE')
      WRITE(6,72)
   72 FORMAT(48X,'DEGREES',3X,'DEGREES',1X,'YARDS  ')
      WRITE(6,10)
      DO 19 I=1,40
      READ(5,4) IEL(I), IAZ(I), IRAN(I)
    4 FORMAT(3I10)
      ITIME(I)=KTIME(I)
      TIME(I)=FLOAT(ITIME(I))/10.
C     THE TIME IS A FUNCTION OF THE NUMBER OF ORDER OF
C     THE CARD, IT GOES FROM 0 TO 9 IN 1 SEC. INCREMENTS
C     AND FROM 9.5 TO 12.4 IN  0.1 SEC. INCREMENTS.
      ELEV(I)=CTDO(IEL(I),1)
```

```
      AZ(I)=CTDO(IAZ(I),1)
      RAN(I)=CTDO(IRAN(I),0)
C     THE CONVERSION FROM THE CODED DATA INTO DECIMAL REAL
C     NUMBER IS ACCOMPLISHED.
      RELEV=ELEV(I)/57.296
      RNM=RAN(I)/2000.
      ALA(I)= RNM  *SIN(RELEV)*6080.
      XAT(I)=TRANS( RNM  ,ELEV(I),AZ(I))*2000.-750.
C     WE NOW HAVE THE ALTITUDE AND DISTANCE TO TARGET.
      WRITE(6,75)IEL(I),IAZ(I),IRAN(I),ELEV(I),AZ(I),RAN(I)
   75 FORMAT(7X,I10,3X,I10,1X,I10,7X,F4.1,6X,F5.1,3X,F6.1)
   19 CONTINUE
      CALL SMOOTH
      VELOC(1)=0.0
      DIVE(1)=0.0
      DO 50 L=2,10
      J=L-1
      SALT=ALTSM(J)
      SXAT=XATSM(J)
      FALT=ALTSM(L)
      FXAT=XATSM(L)
      CALL SPADA
      DIVE(L)=DEG
   50 VELOC(L)=VKTS
      SALT=ALTSM(10)
      SXAT=XATSM(10)
      FALT=ALTSM(11)
      FXAT=XATSM(11)
      CALL SPADA
      DIVE(11)=DEG
      VELOC(11)=VKTS*2.
      DO 55 L=12,40
      J=L-1
      SALT=ALTSM(J)
      SXAT=XATSM(J)
      FALT=ALTSM(L)
      FXAT=XATSM(L)
      CALL SPADA
      DIVE(L)=DEG
   55 VELOC(L)=VKTS*10.
      WRITE(6,9)
      WRITE(6,60)
      GO TO 64
   41 DO 42 I=1,40
      READ(5,43)TIME(I),VELOC(I),DIVE(I),ALTSM(I),XATSM(I)
   43 FORMAT(F4.1,2F5.1,2F8.2)
   42 CONTINUE
   64 WRITE(6,77)
   77 FORMAT(12X,'SMOOTHED DATA FOR THE PL/M PROGRAM.')
      WRITE(6,10)
      WRITE(6,80)
   80 FORMAT(12X,'TIME   SPEED  DIVE ANG  ALT.     DIST TO',
     1' TARGET')
      WRITE(6,81)
   81 FORMAT(12X,'SEC.   KNOTS  DEGREES   FEET     YARDS  ')
      WRITE(6,10)
      DO 99 I=1,40
      WRITE(6,82)TIME(I),VELOC(I),DIVE(I),ALTSM(I),XATSM(I)
   82 FORMAT(12X,F4.1,2X, F5.1,2X,F5.1,5X,F6.1,2X,F6.1)
      WRITE(7,83)TIME(I),VELOC(I),DIVE(I),ALTSM(I),XATSM(I)
   83 FORMAT(F4.1,F5.1,F5.1,F6.1,F6.1)
   99 CONTINUE
      WRITE(6,9)
      WRITE(6,60)
      WRITE(6,88)
   88 FORMAT(12X,'PL/M FLOATING POINT NUMBERS:')
      WRITE(6,10)
      WRITE(6,90)
   90 FORMAT(12X,'TIME  SPEED     DIVE ANG  ALTITUDE  '
     1,'DIST TO TARGET')
      WRITE(6,91)
   91 FORMAT(12X,'SEC.  KNOTS     DEGREES   FEET        '
```

```
      1,'YARDS    ')
       WRITE(6,10)
       DO 22 I=1,40
       IT=ITIME(I)
       CALL STIHA(IT)
       VI=VELOC(I)
       CALL CTHO(VI)
       CALL CTHT
       IW=2
       CALL SIHA(IW)
       DIV=DIVE(I)
       IF (DIV.LT.0.) DIV=-DIV
       CALL CTHO(DIV)
       CALL CTHT
       IF (DIVE(I).LT.0.) IARRE(1)=1
       IW=8
       CALL SIHA(IW)
       AI=ALTSM(I)
       CALL CTHO(AI)
       CALL CTHT
       IW=14
       CALL SIHA(IW)
       XI=XATSM(I)
       CALL CTHO(XI)
       CALL CTHT
       IW=20
       CALL SIHA(IW)
C      NOW WE'RE READY TO PRINT THE PL/M FL. POINT NUMBERS.
       WRITE(6,92)(IHA(J),J=1,26)
   92  FORMAT(12X,A1,A1,4X,A1,A1,1X,A1,A1,1X,A1,A1,2X,A1,A1,
      11X,A1,A1,1X,A1,A1,2X,A1,A1,1X,A1,A1,1X,A1,A1,2X,
      2A1,A1,1X,A1,A1,1X,A1,A1)
C      PUNCH 2 CARDS WITH THE PROPER FORMAT FOR THE PL/M  PGM
       WRITE(7,8)(IHA(K),K=1,12)
   8   FORMAT(12X,'O',A1,A1,'H,','O',A1,A1,'H,','O',A1,A1,
      1 'H,','O',A1,A1,'H,','O',A1,A1,'H,','O',A1,A1,'H,')
       WRITE(7,97)(IHA(K),K=13,26)
   97  FORMAT(12X,'O',A1,A1,'H,','O',A1,A1,'H,','O',A1,A1,
      1 'H,','O',A1,A1,'H,','O',A1,A1,'H,','O',A1,A1,'H,',
      2 'O',A1,A1,'H,')
   22  CONTINUE
   101 CONTINUE
       WRITE(6,9)
       STOP
       END
       SUBROUTINE SMOOTH
C     ***********************************************************
C     *                                                         *
C     *    THIS SUBROUTINE PRODUCES SMOOTHED DATA FROM          *
C     *    THE ORIGINAL DATA, BY MEANS OF A LEAST-SQUARES       *
C     *    FIT USING ORTHOGONAL POLYNOMIALS.                    *
C     *                                                         *
C     ***********************************************************
       COMMON/MAORT/ ALTSM(40), XATSM(40), TIME(40),
      1               ALT(40), XAT(40)
       DIMENSION C(4), F(4)
       S0=40.0
       B=0.0
       DO 8 I=1,40
   8   B=B+TIME(I)
       B0=B/S0
       S1=0.0
       DO 9 I=1,40
   9   S1=S1+(TIME(I)-B0)**2
       B=0.0
       DO 10 I=1,40
   10  B=B+(TIME(I))*(TIME(I)-B0)**2
       B1=B/S1
       C1=S1/S0
       S2=0.0
       DO 11 I=1,40
   11  S2=S2+((TIME(I)-B1)*(TIME(I)-B0)-C1)**2
```

```
      B=0.0
      DO 111 I=1,40
  111 B=B+TIME(I)*((TIME(I)-B1)*(TIME(I)-B0)-C1)**2
      B2=B/S2
      C2=S2/S1
      S3=0.0
      DO 112 I=1,40
  112 S3=S3+((TIME(I)-B2)*((TIME(I)-B1)*(TIME(I)-B0)-C1)-
     1C2*(TIME(I)-B0))**2
      D=0.0
      E=0.0
      DO 12 I=1,40
      D=D+ALT(I)
  12  E=E+XAT(I)
      C(1)=D/S0
      F(1)=E/S0
      D=0.0
      E=0.0
      DO 13 I=1,40
      D=D+(ALT(I))*(TIME(I)-B0)
  13  E=E+(XAT(I))*(TIME(I)-B0)
      C(2)=D/S1
      F(2)=E/S1
      D=0.0
      E=0.0
      DO 14 I=1,40
      D=D+((ALT(I))*((TIME(I)-B1)*(TIME(I)-B0)-C1))
  14  E=E+((XAT(I))*((TIME(I)-B1)*(TIME(I)-B0)-C1))
      C(3)=D/S2
      F(3)=E/S2
      D=0.0
      E=0.0
      DO 141 I=1,40
      D=D+(ALT(I))*((TIME(I)-B2)*((TIME(I)-B1)*(TIME(I)
     1-B0)-C1)-C2*(TIME(I)-B0))
  141 E=E+(XAT(I))*((TIME(I)-B2)*((TIME(I)-B1)*(TIME(I)
     1-B0)-C1)-C2*(TIME(I)-B0))
      C(4)=D/S3
      F(4)=E/S3
      DO 15 I=1,40
      ALTSM(I)=C(1)+C(2)*(TIME(I)-B0)+C(3)*((TIME(I)-B1)*
     1 (TIME(I)-B0)-C1)+C(4)*((TIME(I)-B2)*((TIME(I)-B1)
     2 *(TIME(I)-B0)-C1)-C2*(TIME(I)-B0))
  15  XATSM(I)=F(1)+F(2)*(TIME(I)-B0)+F(3)*((TIME(I)-B1)*
     1 (TIME(I)-B0)-C1)+F(4)*((TIME(I)-B2)*((TIME(I)-B1)
     2 *(TIME(I)-B0)-C1)-C2*(TIME(I)-B0))
      RETURN
      END
      FUNCTION KTIME(I)
C    **************************************************
C    *                                                *
C    *   THIS FUNCTION GENERATES THE TIME ACCORDING   *
C    *   TO THE CARD POSITION.                        *
C    *                                                *
C    **************************************************
      IF (I.LE.10)GO TO 1
      KTIME=(I-11)+95
      GO TO 2
  1   KTIME=(I-1)*10
  2   RETURN
      END
      FUNCTION CTDO(KK,N)
C    **************************************************
C    *                                                *
C    *   THIS FUNCTION CONVERTS FROM THE CODED OCTAL  *
C    *   NUMBER TO DECIMAL NUMBER.                    *
C    *                                                *
C    **************************************************
      K=KK
      IACUM=0
      DO 1 I=1,6
      J=K/(10**(6-I))
```

```
      L=J*(8**(6-I))
      IACUM=IACUM+L
    1 K=K-J*10**(6-I)
C     N=0 MEANS RANGE, N=1,MEANS ELEV OR AZIMUTH.
      IF (N.EQ.1) CFAC=360./65535.
      IF (N.EQ.1) GO TO 2
      CFAC=2.
    2 CTDD=IACUM*CFAC
      RETURN
      END
      FUNCTION COMP(X,J)
C     *********************************************************
C     *                                                       *
C     *   THIS FUNCTION PRODUCES THE IARRF ARRAY WHEN         *
C     *   CALLED BY  THE CTHO SUBROUTINE.                     *
C     *                                                       *
C     *********************************************************
      COMMON/ALL/ IARRI(16),IARRE(8) ,IARRF(16),KI,INC
      X=X-1.
      IARRF(J)=1
      COMP=X
      RETURN
      END
      SUBROUTINE CTHO(X)
C     *********************************************************
C     *                                                       *
C     *   THIS SUBROUTINE BUILDS THE IARRI ARRAY AND,BY       *
C     *   CALLING THE  FUNCTION COMP,THE IARRF ARRAY.         *
C     *   BOTH ARRAYS HAVE 16 ELEMENTS AND REPRESENT          *
C     *   TWO 8 BITS WORDS.                                   *
C     *   THE ARRAY IARRI WILL FINALLY HOLD THE FOUR          *
C     *   HEXADECIMAL CHARACTERS FOR THE MANTISSA OF          *
C     *   THE PL/M FLOATING POINT NUMBER.                     *
C     *                                                       *
C     *********************************************************
      COMMON/ALL/ IARRI(16),IARRE(8) ,IARRF(16),KI,INC
      IPIN=INT(X)
      PFN=X-IPIN
      KI=0
      INC=0
C     INC IS THE NUMBER OF LEADING ZEROES OF THE FRACTION.
C     IT WILL BE USEFUL IF THE NUMBER IS LESS THAN 1.
      DO 2 I=1,16
      IARRI(I)=0
    2 IARRF(I)=0
      IF (IPIN .NE. 0) GO TO 7
      IF (PFN.EQ.0.) GO TO 9
      IF (PFN .LT. .5) INC=1
      IF (PFN .LT. .25) INC=2
      IF (PFN .LT. .125) INC=3
      IF (INC.EQ.0) GO TO 3
      DO 8 I=1,INC
    8 PFN=PFN*2.
      GO TO 3
    7 DO 1 IJ=1,16
      KI=IJ
      JPIN=IPIN
      IPIN=IPIN/2
      IF (MOD(JPIN,2).EQ.1) IARRI(IJ)=1
      IF (IPIN.EQ.0) GO TO 3
    1 CONTINUE
C     16-KI IS THE NUMBER OF PLACES THAT THE ARRAY IARRI
C     HAS  TO BE SHIFTED.
C     THE VALUE OF THE EXPONENT IS KI.
    3 DO 4 J=1,16
      KJ=J
      PFN=PFN*2.
    4 IF(PFN.GT.1.) PFN=COMP(PFN,KJ)
C     SHIFT RIGHT. THE ORIGIN OF THE HEXADECIMAL NUMBER WILL
C     BE THE RIGHT SIDE.
      DO 5 I=1,KI
    5 IARRI(17-I)=IARRI(KI+1-I)
```

```
      KK=16-KI
C     COMPLETE THE 16 BITS USING THE FRACTIONAL PART.
      DO 6 I=1,KK
    6 IARRI(17-KI-I)=IARRF(I)
      RETURN
    9 KI=100
      RETURN
      END
      SUBROUTINE CTHT
C     ***************************************************************
C     *                                                             *
C     *   THIS SUBROUTINE FILLS IN THE BITS  IN THE                 *
C     *   ARRAY IARRE,CORRESPONDING TO THE EXPONENT                 *
C     *   OF THE PL/M FLOATING POINT NUMBER.                        *
C     *                                                             *
C     ***************************************************************
      COMMON/ALL/ IARRI(16),IARRE(8) ,IARRF(16),KI,INC
      IARRE(1)=0
      IARRE(2)=1
      IF (INC .NE. 0) GO TO 4
      DO 2 I=3,8
    2 IARRE(I)=0
      IF (KI .EQ. 100) GO TO 3
      DO 1 IN=1,6
      KK=KI
      KI=KI/2
      IF( MOD(KK,2).EQ.1) IARRE(9-IN)=1
      IF (KI.EQ.0) GO TO 3
    1 CONTINUE
      GO TO 3
    4 IARRE(2)=0
      DO 8 I=3,8
    8 IARRE(I)=1
      GO TO (5,6,7),INC
    5 GO TO 3
    6 IARRE(8)=0
      GO TO 3
    7 IARRE(7)=0
    3 RETURN
      END
      FUNCTION TRANS(R,E,A)
C     ***************************************************************
C     *                                                             *
C     *   THIS FUNCTION TRANSFORMS ELEVATION,AZIMUTH                *
C     *   AND RANGE INTO DISTANCE TO TARGET.                        *
C     *                                                             *
C     ***************************************************************
      ALF=(A-180.)/57.296
      S=SIN(1.5708-ALF)
      C=SIN(72.20528/57.296)
      D=465./(6080.*COS(ALF))
      EL=E/57.296
      TRANS=(R*COS(EL)-D)*S/C
      RETURN
      END
      FUNCTION IHEX(IY)
C     ***************************************************************
C     *                                                             *
C     *   THIS FUNCTION TAKES FOUR ELEMENTS AT A TIME               *
C     *   FROM IARRI AND IARRE TO COMPUTE THE DECIMAL               *
C     *   VALUE OF THE CHARACTER.                                   *
C     *                                                             *
C     ***************************************************************
      COMMON/ALL/ IARRI(16),IARRE(8) ,IARRF(16),KI,INC
      DIMENSION IAUX(4)
      GO TO(1,2,3,4,5,6),IY
    1 DO 7 J=1,4
    7 IAUX(J)=IARRI(17-J)
      GO TO 13
    2 DO 8 J=1,4
    8 IAUX(J)=IARRI(13-J)
      GO TO 13
```

```
  3   DO 9 J=1,4
  9   IAUX(J)=IARRI(9-J)
      GO TO 13
  4   DO 10 J=1,4
  10  IAUX(J)=IARRI(5-J)
      GO TO 13
  5   DO 11 J=1,4
  11  IAUX(J)=IARRE(J)
      GO TO 13
  6   DO 12 J=1,4
  12  IAUX(J)=IARRE(J+4)
  13  IACUM=0
      DO 14 J=1,4
  14  IF(IAUX(J).EQ.1) IACUM=IACUM+2**(4-J)
      IHEX=IACUM
      RETURN
      END
      FUNCTION    KHEX(IZ)
C    ************************************************
C    *                                              *
C    *   THIS FUNCTION RETURNS THE PROPER HEXADECIMAL   *
C    *   CHARACTER CORRESPONDING TO THE DECIMAL VALUE . *
C    *                                              *
C    ************************************************
      COMMON/MAKH/IZERO,ITWO,ITHREE,IFOUR,IFIVE,ISIX,ISEVEN,
     1IEIGHT,ININE,ITEN,IELEVN,ITWELV,ITHIRT,IFOURT,IFIFT,
     1IONE
      KU=IZ+1
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16),KU
  1   KHEX=IZERO
      GO TO 17
  2   KHEX=IONE
      GO TO 17
  3   KHEX=ITWO
      GO TO 17
  4   KHEX=ITHREE
      GO TO 17
  5   KHEX=IFOUR
      GO TO 17
  6   KHEX=IFIVE
      GO TO 17
  7   KHEX=ISIX
      GO TO 17
  8   KHEX=ISEVEN
      GO TO 17
  9   KHEX=IEIGHT
      GO TO 17
  10  KHEX=ININE
      GO TO 17
  11  KHEX=ITEN
      GO TO 17
  12  KHEX=IELEVN
      GO TO 17
  13  KHEX=ITWELV
      GO TO 17
  14  KHEX=ITHIRT
      GO TO 17
  15  KHEX=IFOURT
      GO TO 17
  16  KHEX=IFIFT
  17  RETURN
      END
      SUBROUTINE STIHA(I)
C    ************************************************
C    *                                              *
C    *   THIS SUBROUTINE FILLS THE PROPER ELEMENTS OF   *
C    *   THE ARRAY IHA WITH THE HEXADECIMAL CHARACTERS  *
C    *   THAT STANDS FOR TIME.                          *
C    *                                              *
C    ************************************************
      COMMON/MASH/ IHA(20)
      ISEC=MOD(I,16)
```

```
      KNT=I/16
      IHA(1)=KHEX(KNT)
      IHA(2)=KHEX(ISEC)
      RETURN
      END
      SUBROUTINE SIHA(JOTA)
C    ****************************************************
C    *                                                  *
C    *   THIS SUBROUTINE FILLS THE IHA ARRAY WITH THE   *
C    *   CHARACTERS FOR SPEED, DIVE ANGLE, ALTITUDE     *
C    *   AND DISTANCE TO TARGET.                        *
C    *                                                  *
C    ****************************************************
      COMMON/MASH/ IHA(26)
C    J=2 MEANS SPEED, J=8 MEANS DIVE ANG,J=14 MEANS ALTITUDE
C    AND J=20 MEANS DISTANCE TO TARGET.
      DO 1 I=1,6
      IHV=IHEX(I)
      KHX=KHEX(IHV)
  1   IHA(I+JOTA)=KHX
      RETURN
      END
      SUBROUTINE SPADA
C    ****************************************************
C    *                                                  *
C    *   THIS SUBROUTINE COMPUTES SPEED AND DIVE ANGLE. *
C    *                                                  *
C    ****************************************************
      COMMON/MASD/SALT,SXAT,FALT,FXAT
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
      SF=SALT-FALT
      DALT=ABS(SF)/6080.
      DXAT=(SXAT-FXAT)/2000.
      DRAN=SQRT(DALT*DALT+DXAT*DXAT)
      VKTS=DRAN*3600.
      DAX=DALT/DXAT
      DEG=ATAN(DAX)*57.296
      IF (SALT.GT.FALT) DEG=-DEG
      RETURN
      END
```

```
RUN 1:

INPUT DATA: CODED OCTAL DIGITS      CONVERSION TO DECIMAL:
ELEVATION   AZIMUTH      RANGE      ELEVATION AZIMUTH RANGE
                                    DEGREES   DEGREES YARDS


11401       104206       5305       26.7      192.0   5514.0
11342       104374       5137       26.6      192.6   5310.0
11264       104574       4771       26.3      193.3   5106.0
11207       105013       4611       26.1      194.1   4882.0
11131       105254       4440       25.8      195.0   4672.0
11046       105534       4261       25.5      196.0   4450.0
10762       106040       4104       25.2      197.1   4232.0
10674       106374       3722       24.9      198.3   4004.0
10604       106772       3540       24.6      199.7   3776.0
10510       107435       3356       24.3      201.3   3548.0
10444       107677       3266       24.1      202.1   3436.0
10434       107740       3253       24.1      202.3   3414.0
10424       110002       3241       24.0      202.5   3394.0
10415       110046       3227       24.0      202.7   3374.0
10404       110111       3213       23.9      202.9   3350.0
10374       110156       3202       23.9      203.1   3332.0
10365       110221       3166       23.8      203.3   3308.0
10356       110270       3151       23.8      203.5   3282.0
10347       110340       3140       23.8      203.7   3264.0
10337       110411       3123       23.7      204.0   3238.0
10324       110461       3110       23.7      204.2   3216.0
10311       110525       3074       23.6      204.4   3192.0
10276       110572       3060       23.5      204.6   3168.0
10264       110636       3044       23.5      204.8   3144.0
10252       110705       3033       23.4      205.0   3126.0
10240       110756       3015       23.4      205.2   3098.0
10225       111027       3002       23.3      205.4   3076.0
10212       111100       2766       23.3      205.7   3052.0
10177       111153       2754       23.2      205.9   3032.0
10165       111226       2742       23.1      206.1   3012.0
10151       111302       2727       23.1      206.4   2990.0
10136       111360       2714       23.0      206.6   2968.0
10123       111436       2703       23.0      206.9   2950.0
10111       111515       2663       22.9      207.1   2918.0
10077       111574       2650       22.8      207.4   2896.0
10066       111656       2640       22.8      207.7   2880.0
10053       111734       2624       22.7      207.9   2856.0
10037       112013       2611       22.7      208.2   2834.0
10023       112076       2574       22.6      208.5   2808.0
10006       112161       2563       22.5      208.7   2790.0
```

```
SMOOTHED DATA FOR THE PL/M PROGRAM.

TIME   SPEED   DIVE ANG      ALT.     DIST TO TARGET
SEC.   KNOTS   DEGREES       FEET     YARDS


 0.0     0.0      0.0       7549.4   4148.9
 1.0   398.0    -30.5       7207.8   3958.5
 2.0   406.8    -29.9       6865.5   3762.6
 3.0   415.2    -29.3       6522.3   3561.4
 4.0   423.4    -28.8       6178.3   3355.3
 5.0   431.1    -28.3       5833.3   3144.3
 6.0   438.5    -27.8       5487.4   2928.9
 7.0   445.5    -27.5       5140.5   2709.3
 8.0   452.1    -27.1       4792.5   2485.8
 9.0   458.4    -26.8       4443.4   2258.5
 9.5   462.8    -26.6       4268.4   2143.5
 9.6   464.5    -26.5       4233.4   2120.4
 9.7   465.1    -26.5       4198.4   2097.3
 9.8   465.6    -26.5       4163.3   2074.2
 9.9   466.2    -26.4       4128.3   2051.0
10.0   466.7    -26.4       4093.2   2027.8
10.1   467.3    -26.4       4058.1   2004.5
10.2   467.8    -26.4       4023.0   1981.2
10.3   468.3    -26.4       3987.9   1957.9
10.4   468.9    -26.3       3952.8   1934.6
10.5   469.4    -26.3       3917.6   1911.2
10.6   469.9    -26.3       3882.5   1887.8
10.7   470.4    -26.3       3847.3   1864.3
10.8   471.0    -26.2       3812.1   1840.9
10.9   471.5    -26.2       3776.9   1817.4
11.0   472.0    -26.2       3741.7   1793.9
11.1   472.5    -26.2       3706.5   1770.3
11.2   473.0    -26.2       3671.3   1746.7
11.3   473.5    -26.1       3636.1   1723.1
11.4   474.0    -26.1       3600.8   1699.5
11.5   474.5    -26.1       3565.5   1675.8
11.6   474.9    -26.1       3530.3   1652.1
11.7   475.4    -26.1       3495.0   1628.4
11.8   475.9    -26.1       3459.7   1604.6
11.9   476.4    -26.0       3424.3   1580.8
12.0   476.9    -26.0       3389.0   1557.0
12.1   477.3    -26.0       3353.7   1533.2
12.2   477.8    -26.0       3318.3   1509.3
12.3   478.2    -26.0       3282.9   1485.5
12.4   478.7    -26.0       3247.6   1461.5
```

```
PL/M FLOATING POINT NUMBERS:

TIME   SPEED       DIVE ANG   ALTITUDE    DIST TO TARGET
SEC.   KNOTS       DEGREES    FEET        YARDS


00     00 00 40    00 00 40   EB EB 4D    81 A7 4D
0A     C6 FA 49    F4 62 C5   E1 3E 4D    F7 68 4C
14     CB 62 49    EF 21 C5   D6 8B 4D    EB 29 4C
1E     CF 9E 49    EA 62 C5   CB D2 4D    DE 97 4C
28     D3 AC 49    E6 18 C5   C1 11 4D    D1 B4 4C
32     D7 8D 49    E2 3C C5   B6 4A 4D    C4 85 4C
3C     DB 3E 49    DE C4 C5   AB 7B 4D    B7 0F 4C
46     DE BE 49    DB A8 C5   A0 A3 4D    A9 55 4C
50     E2 0E 49    D8 E4 C5   95 C4 4D    9B 5C 4C
5A     E5 2C 49    D6 70 C5   8A D8 4D    8D 27 4C
5F     E7 63 49    D4 C8 C5   85 63 4D    85 F8 4C
60     E8 3D 49    D4 29 C5   84 4B 4D    84 87 4C
61     E8 88 49    D4 00 C5   83 32 4D    83 15 4C
62     E8 CD 49    D3 C4 C5   82 1A 4D    81 A2 4C
63     E9 15 49    D3 97 C5   81 02 4D    80 2F 4C
64     E9 59 49    D3 61 C5   FF D2 4C    FD 78 4B
65     E9 A0 49    D3 36 C5   FD A1 4C    FA 90 4B
66     E9 E5 49    D3 07 C5   FB 6F 4C    F7 A7 4B
67     EA 2B 49    D2 D8 C5   F9 3E 4C    F4 BD 4B
68     EA 6F 49    D2 AA C5   F7 0C 4C    F1 D2 4B
69     EA B2 49    D2 7E C5   F4 D9 4C    EE E6 4B
6A     EA F5 49    D2 51 C5   F2 A7 4C    EB F9 4B
6B     EB 38 49    D2 25 C5   F0 74 4C    E9 0B 4B
6C     EB 7A 49    D1 FA C5   EE 41 4C    E6 1C 4B
6D     EB BC 49    D1 D0 C5   EC 0E 4C    E3 2C 4B
6E     EB FD 49    D1 A5 C5   E9 DB 4C    E0 3B 4B
6F     EC 3D 49    D1 7D C5   E7 A8 4C    DD 49 4B
70     EC 7E 49    D1 54 C5   E5 74 4C    DA 57 4B
71     EC BE 49    D1 2C C5   E3 40 4C    D7 63 4B
72     EC FD 49    D1 05 C5   E1 0C 4C    D4 6E 4B
73     ED 3B 49    D0 DF C5   DE D8 4C    D1 79 4B
74     ED 79 49    D0 BA C5   DC A4 4C    CE 83 4B
75     ED B7 49    D0 94 C5   DA 6F 4C    CB 8B 4B
76     ED F4 49    D0 6F C5   D8 3A 4C    C8 93 4B
77     EE 31 49    D0 4B C5   D6 05 4C    C5 9A 4B
78     EE 6D 49    D0 28 C5   D3 D0 4C    C2 A1 4B
79     EE A8 49    D0 06 C5   D1 9A 4C    BF A6 4B
7A     EE E3 49    CF E4 C5   CF 65 4C    BC AA 4B
7B     EF 1D 49    CF C2 C5   CD 2F 4C    B9 AE 4B
7C     EF 58 49    CF A1 C5   CA F9 4C    B6 B1 4B
```

```
RUN 2:

INPUT DATA: CODED OCTAL DIGITS          CONVERSION TO DECIMAL:
ELEVATION   AZIMUTH      RANGE          ELEVATION AZIMUTH RANGE
                                        DEGREES   DEGREES YARDS


13621       103634       5333           33.1      190.7   5558.0
13543       103710       5177           32.9      190.9   5374.0
13460       104056       5045           32.6      191.5   5194.0
13365       104255       4706           32.3      192.2   5004.0
13311       104464       4540           32.0      192.9   4800.0
13227       104732       4377           31.8      193.9   4606.0
13140       105175       4227           31.5      194.8   4398.0
13025       105510       4060           31.1      195.9   4192.0
12703       106051       3713           30.6      197.1   3990.0
12555       106444       3532           30.1      198.5   3764.0
12500       106664       3446           29.9      199.3   3660.0
12467       106722       3431           29.8      199.4   3634.0
12456       106755       3422           29.8      199.6   3620.0
12446       107011       3406           29.7      199.7   3596.0
12436       107043       3372           29.7      199.9   3572.0
12426       107101       3357           29.7      200.0   3550.0
12416       107141       3345           29.6      200.2   3530.0
12403       107203       3326           29.5      200.4   3500.0
12367       107243       3320           29.5      200.6   3488.0
12353       107301       3303           29.4      200.8   3462.0
12341       107341       3267           29.4      200.9   3438.0
12330       107402       3256           29.3      201.1   3420.0
12320       107446       3245           29.3      201.3   3402.0
12310       107511       3231           29.2      201.5   3378.0
12277       107555       3214           29.2      201.7   3352.0
12267       107620       3202           29.1      201.9   3332.0
12256       107663       3165           29.1      202.1   3306.0
12246       107726       3160           29.0      202.3   3296.0
12236       107773       3145           29.0      202.5   3274.0
12225       110043       3123           28.9      202.7   3238.0
12214       110115       3114           28.9      202.9   3224.0
12203       110166       3076           28.8      203.2   3196.0
12172       110235       3062           28.8      203.4   3172.0
12161       110305       3050           28.7      203.6   3152.0
12150       110356       3036           28.7      203.8   3132.0
12141       110431       3023           28.7      204.0   3110.0
12131       110504       3010           28.6      204.3   3088.0
12121       110557       2777           28.6      204.5   3070.0
12111       110634       2761           28.5      204.8   3042.0
12102       110711       2753           28.5      205.0   3030.0
```

```
SMOOTHED DATA FOR THE PL/M PROGRAM.

TIME   SPEED   DIVE ANG    ALT.     DIST TO TARGET
SEC.   KNOTS   DEGREES     FEET     YARDS


 0.0     0.0      0.0      9225.4   3896.0
 1.0   343.6   -37.0       8876.3   3743.5
 2.0   355.2   -37.2       8513.7   3586.3
 3.0   366.5   -37.2       8139.5   3424.1
 4.0   377.5   -37.1       7755.3   3256.7
 5.0   388.1   -36.8       7362.9   3084.0
 6.0   398.4   -36.4       6964.1   2905.8
 7.0   408.4   -35.8       6560.6   2721.8
 8.0   418.1   -35.1       6154.1   2531.9
 9.0   427.6   -34.4       5746.5   2335.8
 9.5   434.6   -33.7       5542.7   2235.4
 9.6   437.4   -33.5       5502.0   2215.1
 9.7   438.3   -33.4       5461.3   2194.8
 9.8   439.2   -33.3       5420.6   2174.4
 9.9   440.1   -33.2       5379.9   2153.9
10.0   441.0   -33.1       5339.3   2133.4
10.1   441.9   -33.0       5298.7   2112.8
10.2   442.8   -32.9       5258.1   2092.1
10.3   443.8   -32.8       5217.5   2071.4
10.4   444.7   -32.7       5177.0   2050.6
10.5   445.6   -32.6       5136.5   2029.8
10.6   446.5   -32.5       5096.0   2008.8
10.7   447.4   -32.4       5055.5   1987.8
10.8   448.3   -32.3       5015.1   1966.8
10.9   449.2   -32.2       4974.7   1945.6
11.0   450.1   -32.0       4934.4   1924.5
11.1   451.0   -31.9       4894.1   1903.2
11.2   451.9   -31.8       4853.8   1881.9
11.3   452.8   -31.7       4813.6   1860.5
11.4   453.7   -31.6       4773.5   1839.0
11.5   454.6   -31.5       4733.3   1817.5
11.6   455.5   -31.4       4693.3   1795.9
11.7   456.4   -31.3       4653.2   1774.2
11.8   457.3   -31.2       4613.3   1752.5
11.9   458.2   -31.1       4573.4   1730.7
12.0   459.1   -30.9       4533.5   1708.8
12.1   460.0   -30.8       4493.7   1686.8
12.2   460.9   -30.7       4453.9   1664.8
12.3   461.8   -30.6       4414.3   1642.7
12.4   462.7   -30.5       4374.6   1620.6
```

```
PL/M FLOATING PCINT NUMBERS:

TIME   SPEED        DIVE ANG   ALTITUDE   DIST TO TARGET
SEC.   KNOTS        DEGREES    FEET       YARDS


00     00 00 40     00 00 40   90 25 4E   F3 80 4C
0A     AB CF 49     93 F2 C6   8A B1 4E   E9 F8 4C
14     B1 9F 49     94 B8 C6   85 06 4E   E0 24 4C
1E     B7 42 49     94 CA C6   FE 5B 4D   D6 01 4C
28     BC BB 49     94 3B C6   F2 5A 4D   CB 8B 4C
32     C2 08 49     93 17 C6   E6 17 4D   C0 C0 4C
3C     C7 2E 49     91 69 C6   D9 A0 4D   B5 9C 4C
46     CC 2E 49     8F 3A C6   CD 04 4D   AA 1C 4C
50     D1 0B 49     8C 94 C6   C0 51 4D   9E 3E 4C
5A     D5 CA 49     89 7C C6   B3 93 4D   91 FC 4C
5F     D9 49 49     86 E7 C6   AD 35 4D   8B B6 4C
60     DA AD 49     85 D0 C6   AB EF 4D   8A 72 4C
61     DB 22 49     85 70 C6   AA AA 4D   89 2C 4C
62     DB 97 49     85 0D C6   A9 64 4D   87 E6 4C
63     DC 0C 49     84 AB C6   A8 1F 4D   86 9E 4C
64     DC 82 49     84 49 C6   A6 DA 4D   85 56 4C
65     DC F5 49     83 E2 C6   A5 95 4D   84 0C 4C
66     DD 6A 49     83 7D C6   A4 50 4D   82 C2 4C
67     DD E1 49     83 1C C6   A3 0C 4D   81 76 4C
68     DE 55 49     82 82 C6   A1 C7 4D   80 29 4C
69     DE C8 49     82 49 C6   A0 83 4D   FD B8 4B
6A     DF 3D 49     81 E3 C6   9F 3F 4D   FB 1A 4B
6B     DF B1 49     81 77 C6   9D FC 4D   F8 7A 4B
6C     E0 24 49     81 0D C6   9C B8 4D   F5 D8 4B
6D     E0 9A 49     80 A0 C6   9B 75 4D   F3 34 4B
6E     E1 08 49     80 2F C6   9A 33 4D   F0 8E 4B
6F     E1 80 49     FF 8D C5   98 F0 4D   ED E6 4B
70     E1 F1 49     FE A9 C5   97 AE 4D   EB 3B 4B
71     E2 68 49     FD D5 C5   96 6C 4D   E8 8E 4B
72     E2 D9 49     FC EC C5   95 2B 4D   E5 E0 4B
73     E3 4C 49     FC 09 C5   93 EA 4D   E3 2F 4B
74     E3 BE 49     FB 22 C5   92 AA 4D   E0 7B 4B
75     E4 33 49     FA 3F C5   91 69 4D   DD C6 4B
76     E4 A4 49     F9 54 C5   90 2A 4D   DB 0E 4B
77     E5 17 49     F8 68 C5   8E EA 4D   D8 54 4B
78     E5 8A 49     F7 7E C5   8D AB 4D   D5 98 4B
79     E5 FE 49     F6 93 C5   8C 6D 4D   D2 DA 4B
7A     E6 6F 49     F5 9F C5   8B 2F 4D   D0 1A 4B
7B     E6 E3 49     F4 B0 C5   89 F2 4D   CD 57 4B
7C     E7 56 49     F3 BD C5   88 B5 4D   CA 92 4B
```

```
RUN 3:

SMOOTHED DATA FOR THE PL/M PROGRAM.

TIME   SPEED   DIVE ANG   ALT.     DIST TO TARGET
SEC.   KNOTS   DEGREES    FEET     YARDS


 0.0     0.0     0.0     2000.0   3699.4
 1.0   300.0     0.0     2000.0   3532.7
 2.0   300.0     0.0     2000.0   3366.0
 3.0   300.0     0.0     2000.0   3199.4
 4.0   300.0     0.0     2000.0   3032.7
 5.0   300.0     0.0     2000.0   2866.0
 6.0   300.0     0.0     2000.0   2699.4
 7.0   300.0     0.0     2000.0   2532.7
 8.0   300.0     0.0     2000.0   2366.0
 9.0   300.0     0.0     2000.0   2199.4
 9.5   300.0     0.0     2000.0   2116.0
 9.6   300.0     0.0     2000.0   2099.4
 9.7   300.0     0.0     2000.0   2082.7
 9.8   300.0     0.0     2000.0   2066.0
 9.9   300.0     0.0     2000.0   2049.4
10.0   300.0     0.0     2000.0   2032.7
10.1   300.0     0.0     2000.0   2016.0
10.2   300.0     0.0     2000.0   1999.4
10.3   300.0     0.0     2000.0   1982.7
10.4   300.0     0.0     2000.0   1966.0
10.5   300.0     0.0     2000.0   1949.4
10.6   300.0     0.0     2000.0   1932.7
10.7   300.0     0.0     2000.0   1916.0
10.8   300.0     0.0     2000.0   1899.4
10.9   300.0     0.0     2000.0   1882.7
11.0   300.0     0.0     2000.0   1866.0
11.1   300.0     0.0     2000.0   1849.4
11.2   300.0     0.0     2000.0   1832.7
11.3   300.0     0.0     2000.0   1816.0
11.4   300.0     0.0     2000.0   1799.4
11.5   300.0     0.0     2000.0   1782.7
11.6   300.0     0.0     2000.0   1766.0
11.7   300.0     0.0     2000.0   1749.4
11.8   300.0     0.0     2000.0   1732.7
11.9   300.0     0.0     2000.0   1716.0
12.0   300.0     0.0     2000.0   1699.4
12.1   300.0     0.0     2000.0   1699.4
12.2   300.0     0.0     2000.0   1666.0
12.3   300.0     0.0     2000.0   1649.4
12.4   300.0     0.0     2000.0   1632.7
```

```
PL/M FLOATING PCINT NUMBERS:

TIME   SPEED       DIVE ANG   ALTITUDE    DIST TO TARGET
SEC.   KNOTS       DEGREES    FEET        YARDS


00     00 00 40    00 00 40   FA 00 4B    E7 35 4C
0A     96 00 49    00 00 40   FA 00 4B    DC CB 4C
14     96 00 49    00 00 40   FA 00 4B    D2 60 4C
1E     96 00 49    00 00 40   FA 00 4B    C7 F5 4C
28     96 00 49    00 00 40   FA 00 4B    BD 8B 4C
32     96 00 49    00 00 40   FA 00 4B    B3 20 4C
3C     96 00 49    00 00 40   FA 00 4B    A8 B5 4C
46     96 00 49    00 00 40   FA 00 4B    9E 4B 4C
50     96 00 49    00 00 40   FA 00 4B    93 E0 4C
5A     96 00 49    00 00 40   FA 00 4B    89 75 4C
5F     96 00 49    00 00 40   FA 00 4B    84 40 4C
60     96 00 49    00 00 40   FA 00 4B    83 35 4C
61     96 00 49    00 00 40   FA 00 4B    82 2B 4C
62     96 00 49    00 00 40   FA 00 4B    81 20 4C
63     96 00 49    00 00 40   FA 00 4B    80 15 4C
64     96 00 49    00 00 40   FA 00 4B    FE 16 4B
65     96 00 49    00 00 40   FA 00 4B    FC 00 4B
66     96 00 49    00 00 40   FA 00 4B    F9 EB 4B
67     96 00 49    00 00 40   FA 00 4B    F7 D6 4B
68     96 00 49    00 00 40   FA 00 4B    F5 C0 4B
69     96 00 49    00 00 40   FA 00 4B    F3 AB 4B
6A     96 00 49    00 00 40   FA 00 4B    F1 96 4B
6B     96 00 49    00 00 40   FA 00 4B    EF 81 4B
6C     96 00 49    00 00 40   FA 00 4B    ED 6B 4B
6D     96 00 49    00 00 40   FA 00 4B    EB 56 4B
6E     96 00 49    00 00 40   FA 00 4B    E9 41 4B
6F     96 00 49    00 00 40   FA 00 4B    E7 2B 4B
70     96 00 49    00 00 40   FA 00 4B    E5 16 4B
71     96 00 49    00 00 40   FA 00 4B    E3 01 4B
72     96 00 49    00 00 40   FA 00 4B    E0 EB 4B
73     96 00 49    00 00 40   FA 00 4B    DE D6 4B
74     96 00 49    00 00 40   FA 00 4B    DC C1 4B
75     96 00 49    00 00 40   FA 00 4B    DA AB 4B
76     96 00 49    00 00 40   FA 00 4B    D8 96 4B
77     96 00 49    00 00 40   FA 00 4B    D6 81 4B
78     96 00 49    00 00 40   FA 00 4B    D4 6B 4B
79     96 00 49    00 00 40   FA 00 4B    D4 6B 4B
7A     96 00 49    00 00 40   FA 00 4B    D0 41 4B
7B     96 00 49    00 00 40   FA 00 4B    CE 2B 4B
7C     96 00 49    00 00 40   FA 00 4B    CC 16 4B
```

```
RUN 4:

SMOOTHED DATA FOR THE PL/M PROGRAM.

TIME   SPEED  DIVE ANG    ALT.     DIST TO TARGET
SEC.   KNOTS  DEGREES     FEET     YARDS


 0.0     0.0    0.0       4451.7   5228.2
 1.0   450.0  -10.0       4319.7   4982.0
 2.0   450.0  -10.0       4187.7   4735.8
 3.0   450.0  -10.0       4055.8   4489.6
 4.0   450.0  -10.0       3923.8   4243.4
 5.0   450.0  -10.0       3791.8   3997.2
 6.0   450.0  -10.0       3659.8   3751.0
 7.0   450.0  -10.0       3527.9   3504.8
 8.0   450.0  -10.0       3395.9   3258.6
 9.0   450.0  -10.0       3263.9   3012.4
 9.5   450.0  -10.0       3198.0   2889.3
 9.6   450.0  -10.0       3184.8   2864.7
 9.7   450.0  -10.0       3171.6   2840.1
 9.8   450.0  -10.0       3158.4   2815.4
 9.9   450.0  -10.0       3145.2   2790.8
10.0   450.0  -10.0       3132.0   2766.2
10.1   450.0  -10.0       3118.8   2741.6
10.2   450.0  -10.0       3105.6   2717.0
10.3   450.0  -10.0       3092.4   2692.3
10.4   450.0  -10.0       3079.2   2667.7
10.5   450.0  -10.0       3066.0   2643.1
10.6   450.0  -10.0       3052.8   2618.5
10.7   450.0  -10.0       3039.6   2593.9
10.8   450.0  -10.0       3026.4   2569.2
10.9   450.0  -10.0       3013.2   2544.6
11.0   450.0  -10.0       3000.0   2520.0
11.1   450.0  -10.0       2986.8   2495.4
11.2   450.0  -10.0       2973.6   2470.8
11.3   450.0  -10.0       2960.4   2446.1
11.4   450.0  -10.0       2947.2   2421.5
11.5   450.0  -10.0       2934.0   2396.9
11.6   450.0  -10.0       2920.8   2372.3
11.7   450.0  -10.0       2907.6   2347.7
11.8   450.0  -10.0       2894.4   2323.0
11.9   450.0  -10.0       2881.2   2298.4
12.0   450.0  -10.0       2868.0   2273.8
12.1   450.0  -10.0       2854.8   2249.2
12.2   450.0  -10.0       2841.6   2224.6
12.3   450.0  -10.0       2828.4   2199.9
12.4   450.0  -10.0       2815.2   2175.3
```

```
PL/M FLOATING PCINT NUMBERS:

TIME   SPEED          DIVE ANG    ALTITUDE    DIST TO TARGET
SEC.   KNOTS          DEGREES     FEET        YARDS


00     00 00 40       C0 00 40    8B 1D 4D    A3 61 4D
0A     E1 00 49       A0 00 C4    86 FD 4D    9B B0 4D
14     E1 00 49       A0 00 C4    82 DD 4D    93 FE 4D
1E     E1 00 49       A0 00 C4    FD 7C 4C    8C 4C 4D
28     E1 00 49       A0 00 C4    F5 3C 4C    84 9B 4D
32     E1 00 49       A0 00 C4    EC FD 4C    F9 D3 4C
3C     E1 00 49       A0 00 C4    E4 BD 4C    EA 70 4C
46     E1 00 49       A0 00 C4    DC 7E 4C    DB 0C 4C
50     E1 00 49       A0 00 C4    D4 3E 4C    CB A9 4C
5A     E1 00 49       A0 00 C4    CB FF 4C    BC 46 4C
5F     E1 00 49       A0 00 C4    C7 DF 4C    B4 94 4C
60     E1 00 49       A0 00 C4    C7 0C 4C    B3 0A 4C
61     E1 00 49       A0 00 C4    C6 38 4C    B1 80 4C
62     E1 00 49       A0 00 C4    C5 65 4C    AF F7 4C
63     E1 00 49       A0 00 C4    C4 92 4C    AE 6D 4C
64     E1 00 49       A0 00 C4    C3 BF 4C    AC E3 4C
65     E1 00 49       A0 00 C4    C2 EC 4C    AB 59 4C
66     E1 00 49       A0 00 C4    C2 19 4C    A9 CF 4C
67     E1 00 49       A0 00 C4    C1 46 4C    A8 45 4C
68     E1 00 49       A0 00 C4    C0 72 4C    A6 BB 4C
69     E1 00 49       A0 00 C4    BF 9F 4C    A5 31 4C
6A     E1 00 49       A0 00 C4    BE CC 4C    A3 A7 4C
6B     E1 00 49       A0 00 C4    BD F9 4C    A2 1D 4C
6C     E1 00 49       A0 00 C4    BD 26 4C    A0 93 4C
6D     E1 00 49       A0 00 C4    BC 53 4C    9F 09 4C
6E     E1 00 49       A0 00 C4    BB 80 4C    9D 80 4C
6F     E1 00 49       A0 00 C4    BA AC 4C    9B F6 4C
70     E1 00 49       A0 00 C4    B9 D9 4C    9A 6C 4C
71     E1 00 49       A0 00 C4    B9 06 4C    98 E2 4C
72     E1 00 49       A0 00 C4    B8 33 4C    97 58 4C
73     E1 00 49       A0 00 C4    B7 60 4C    95 CE 4C
74     E1 00 49       A0 00 C4    B6 8D 4C    94 44 4C
75     E1 00 49       A0 00 C4    B5 B9 4C    92 BA 4C
76     E1 00 49       A0 00 C4    B4 E6 4C    91 30 4C
77     E1 00 49       A0 00 C4    B4 13 4C    8F A6 4C
78     E1 00 49       A0 00 C4    B3 40 4C    8E 1C 4C
79     E1 00 49       A0 00 C4    B2 6D 4C    8C 92 4C
7A     E1 00 49       A0 00 C4    B1 9A 4C    8B 08 4C
7B     E1 00 49       A0 00 C4    B0 C7 4C    89 7F 4C
7C     E1 00 49       A0 00 C4    AF F3 4C    87 F5 4C
```

APPENDIX B

THE EXTRAPOLATING TESTING PROGRAM AND OUTPUT

This appendix shows the program that was used to test the extrapolation routine on the System /360, together with its output.

The data used in this program is the same used in the PL/M implementation.

```
C    ******************************************************
C    *                                                    *
C    *   PROGRAM TO TEST THE EXTRAPOLATION ROUTINE        *
C    *   AND THE SEARCH FOR THE RELEASE  POINT            *
C    *                                                    *
C    *   DTT: DISTANCE TO TARGET IN YARDS                 *
C    *   DRT: DOWN RANGE TRAVEL IN YARDS                  *
C    *                                                    *
C    ******************************************************
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
      COMMON/TIM/RTIME(4),DRO(4),DTT(4)
      DIMENSION VEL(4),DIV(4),ALA(4)
      DIMENSION  TIME(40),VELOC(40),DIVE(40),ALTIT(40)
      DIMENSION  XAC(40),DRT(40)
      YT=0.
      IDNO=10
      WRITE(6,10)
      DO 20 IRUN=1,4
      WRITE(6,10)
  10  FORMAT('1')
      WRITE(6,30)
  30  FORMAT(/////)
      WRITE(6,9)IRUN,IDNO
  9   FORMAT(12X, 'RUN NUMBER',I2,'  IDNO=',I2)
      DO 1 I=1,40
  1   READ(5,2) TIME(I),VELOC(I),DIVE(I),ALTIT(I),XAC(I)
  2   FORMAT(F4.1,F5.1,F5.1,F6.1,F6.1)
C     COMPUTATION OF DOWNRANGE TRAVEL FOR EACH POSITION.
      DO 8 I=2,40
      VKTS=VELOC(I)
      ALT=ALTIT(I)
      DEG=DIVE(I)
      CALL BAL
  8   DRT(I)=X*2000./6080.
      WRITE(6,7)
  7   FORMAT(//)
      WRITE(6,14)
  14  FORMAT(12X,'DATA:')
      WRITE(6,11)
      WRITE(6,15)
  15  FORMAT(12X,'TIME  VKTS   DEG    ALT    DTT    DRT')
      DRT(1)=0.
      DO 12 I=1,40
      WRITE(6,13)TIME(I),VELOC(I),DIVE(I),ALTIT(I),XAC(I),
     1DRT(I)
  13  FORMAT(12X,F4.1,F6.1,F6.1,F7.1,F7.1,F7.1)
  12  CONTINUE
      DO 3 I=1,4
      RTIME(I)= TIME(I*2)
      VEL(I)=VELOC(I*2)
      DIV(I)=DIVE(I*2)
      ALA(I)=ALTIT(I*2)
      DRO(I)=DRT(I*2)
  3   DTT(I)=XAC(I*2)
C     EXTRAPOLATION.
      DO 4 I=1,3
      XRT=10.+FLOAT(I-1)
      XVEL=ORTFIT(XRT,VEL)
      XDIV=ORTFIT(XRT,DIV)
      XALA=ORTFIT(XRT,ALA)
      XDTT=ORTFIT(XRT,DTT)
      XDRO=ORTFIT(XRT,DRO)
      WRITE(6,11)
  11  FORMAT(/)
      WRITE(6,5)XRT,XVEL,XDIV,XALA,XDTT,XDRO
  5   FORMAT(12X,
     1        'EXTRAPOLATION AT TIME ',F4.1,':',
     2/,12X, 'VKTS=',F5.1,' DEG=',F5.1,' ALT=',F6.1,' DTT=',
     3F6.1,  ' DRT=',F6.1)
  4   CONTINUE
      WRITE(6,7)
```

```
      CALL INTER(XD,TR)
      WRITE(6,40)XD
   40 FORMAT(12X,'DOWNRANGE TRAVEL FOUND BY EXTRAPOLATION=',
     1F7.1)
      WRITE(6,41)TR
   41 FORMAT(12X,'TIME OF RELEASE FOUND BY EXTRAPOLATION=',
     1F5.2)
   20 CCNTINUE
      WRITE(6,10)
      STCP
      END
      SUBRCUTINE BAL
C    *************************************************
C    *                                               *
C    *   THIS SUBROUTINE CALCULATES TIME OF FLIGHT   *
C    *   AND DOWN RANGE TRAVEL OF THE BOMBS.         *
C    *                                               *
C    *************************************************
      CCMMCN/AREA/ CT(2,2), CC(3,3,2), IREF,
     1CFORM1, DKG1, DMAX, ITYPE, IBOTH, DM1,CFORM2, DM2,
     2DKG2, DS, SL, FN, VMUZ, RAD, FRACT, VE, THETA, U, DEL,
     3V, VXA, VYA, CF, DM, DKG, VX, VY, TH, Y, YA,
     4VXO, YO, VYO, AN1, AN2, AP1, AP2, IREG, RHO, DTI, DTV
     5    ,G, A, AA,VYK,D,CKDG,MSTG
      COMMON/TAD/T,X
      CCMMON/ACT/VKTS,ALT,DEG,YT,IDNO
C CALL SETDAT TO INITIALIZE THE CONSTANTS
      CALL SETDAT
C CALL DECODE TO INITIALIZE THE BOMB COEFFICIENTS
      CALL DECODE
C CALL TRAJ TO CALCULATE THE SOLUTION TO THE DIFF. EQUATIONS
      CALL TRAJ
      RETURN
      END
      SUBROUTINE SETDAT
      CCMMON/AREA/ CT(2,2), CC(3,3,2), IREF,
     1CFORM1, DKG1, DMAX, ITYPE, IBOTH, DM1,CFORM2, DM2,
     2DKG2, DS, SL, FN, VMUZ, RAD, FRACT, VE, THETA, U, DEL,
     3V, VXA, VYA, CF, DM, DKG, VX, VY, TH, Y, YA,
     4VXO, YO, VYO, AN1, AN2, AP1, AP2, IREG, RHO, DTI, DTV
     5    ,G, A, AA,VYK,D,CKDG,MSTG
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
      G= 32.174
      RAD= 0.0174533
      A= 0.7
      AA= 0.5/A
      VYK = -5.0
      FRACT= 0.5
      DS = 0.0
      CFORM1= 0.0
      CFCRM2= 0.0
      DM1= 0.0
      DM2= 0.0
      DKG1= 0.0
      DKG2= 0.0
      VMUZ= 0.0
      VE= 0.0
      SL= 0.0
      DMAX= 5.0
      ITYPE= -1
      FN= 0.0
      IBOTH= 1
      T= 0.0
      U= VKTS     *1.6878
      DEL= ATAN2(VE,U)
      V= SQRT(U*U+VE*VE)
      THETA= DEG      *RAD
      VXA= (V+VMUZ)* COS(THETA-DEL)
      VYA= (V+VMUZ)* SIN(THETA-DEL)
      RETURN
      END
```

```
      SUBROUTINE DECODE
      COMMON/AREA/ CT(2,2), CC(3,3,2), IREF,
     1CFORM1, DKG1, DMAX, ITYPE, IBOTH, DM1,CFORM2, DM2,
     2DKG2, DS, SL, FN, VMUZ, RAD, FRACT, VS, THETA, U, DEL,
     3V, VXA, VYA, CF, DM, DKG, VX, VY, TH, Y, YA,
     4VXO, YO, VYO, AN1, AN2, AP1, AP2, IREG, RHO, DTI, DTV
     5    ,G, A, AA,VYK,D,CKDG,MSTG
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19
     1,20,21,22,23,24,25,26,27,28), IDNO
C WEAPON CONSTANTS FOR THE MK 43 UNRETARDED
    1 IREF= 4
      DKG1= 2.5506E-03
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 57 UNRETARDED
    2 IREF= 4
      DKG1= 6.2994E-03
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 61 UNRETARDED
    3 IREF= 4
      DKG1= 4.01E-03
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 116 WETEYE
    4 IREF= 2
      DMAX = 3.0
      CFORM1= 3.9235E-03
      DKG1= 2.754E-03
      DTI = 2.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 76 WITH LUG
    5 IREF= 2
      DMAX = 3.0
      CFORM1= 3.9077E-03
      DKG1= 6.3648E-03
      DTI = 1.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 77 FIREBOMB
    6 IREF= 4
      DMAX = 2.
      DKG1= 0.021266
      DTI = 1.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 81
    7 IREF= 1
      CFORM1= 2.5704
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 81 SNAKEYE UNRETARDED
    8 IREF= 4
      DMAX = 3.0
      DKG1= 9.767E-03
      DTI = 2.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 82 MECH FUZE
    9 IREF= 1
      CFORM1= 2.064
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 82 ELEC FUZE
   10 IREF= 1
      CFORM1= 1.4932
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 83 MECH FUZE
   11 IREF= 1
      CFORM1= 1.3431
      DTI = 1.
      GO TO 31
```

```
C WEAPON CONSTANTS FOR THE MK83 ELEC FUZE
   12 IREF= 1
      CFORM1= 1.21
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 84
   13 IREF= 1
      CFORM1= 1.0
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 117 AL
   14 IREF= 1
      CFORM1= 3.12
      DKG1= -1.223E-03
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 86 WET SAND FILLED
   15 IREF= 1
      DMAX = 3.
      CFORM1= 3.4972
      DTI = 2.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 88 WET SAND FILLED
   16 IREF=1
      CFORM1= 1.605
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 82 SNAKEYE UNRETARDED
   17 IREF= 4
      DMAX = 3.
      DKG1= 7.329E-03
      DTI = 1.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 82 SNAKEYE RETARDED
   18 IREF= 1
      ITYPE= 1
      IBOTH= 2
      DKG1= 7.329E-03
      CFORM2= 1.6895E-02
      DM2= 0.38
      DKG2= 0.17166
      DS= 0.6617
      SL= -0.000269
      DTI = 2.0
      GO TO 31
C WEAPON CONSTANTS FOR THE SADEYE T1 = 4.0
   19 IREF= 1
      ITYPE= 1
      IBOTH= 2
      CFORM1= 2.0754
      CFORM2= 0.2217
      DS = 4.267
      DTI = 1.5
      GO TO 31
C WEAPON CONSTANTS FOR THE ROCKEYE II T1 = 4.0
   20 IREF= 1
      ITYPE= 1
      IBOTH= 2
      CFORM1= 2.2973
      DM1= 0.32
      DKG1= 8.175E-03
      CFORM2= 1.1136E-02
      DM2= 0.41
      DKG2= 0.16985
      DS = 4.06
      DTI = 2.0
      GO TO 31
C WEAPON CONSTANTS FOR THE CBU T1 = 4.0
   21 IREF= 1
      ITYPE= 1
      IBOTH= 2
      CFORM1= 2.2404
```

```
      CFORM2= 0.1178
      DS = 4.0
      DTI = 1.62
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 81 SNAKEYE RETARDED
   22 IREF= 1
      ITYPE= 1
      IBOTH= 2
      DKG1= 9.767E-03
      CFORM2= 2.30625E-02
      DM2= 0.38
      DKG2= 0.23287
      DS= 0.679
      SL= -0.000303
      DTI = 1.622
      GO TO 31
C WEAPON CONSTANTS FOR THE GUN
   23 IREF= 3
      DMAX = 1.5
      CFORM1= 2.9964
      DKG1= -0.014992
      VMUZ= 3300.0
      DTI = 0.5
      GO TO 31
C WEAPON CONSTANTS FOR THE ROCKETS
   24 IREF= 3
      ITYPE= 2
      CFORM1= 0.82
      CFORM2 = 1.0
      FN= 1746.0
      DS = 1.4225
      DTI = 1.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 43 RETARDED 0.4 SEC DELAY
   25 IREF= 4
      ITYPE= 0
      DS= 0.98
      DKG2= 1.48
      DTI = 0.31
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 57 RETARDED 0.8 SEC DELAY
   26 IREF= 4
      ITYPE= 0
      DS= 0.89
      DKG2= 2.0
      DTI = 0.22
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 61 RETARDED 0.6 SEC DELAY
   27 IREF= 4
      ITYPE= 0
      DS= 0.89
      DKG2= 2.70
      DTI = 0.1
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 106 MOD 2
   28 IREF= 2
      ITYPE= 2
      CFORM1= 0.1514
      CFORM2= 0.1514
      DS = 0.5
      DTI = 0.8
C SET THE REFERENCE DRAG CURVE COEFFICIENTS AND CUTS
   31 GO TO (32,33,34,51), IREF
   32 CC(1,1,1)= 1.572924E-03
      CC(1,2,1)= 0.0
      CC(1,3,1)= 0.0
      CC(2,1,1)= 4.673409E-02
      CC(2,2,1)=-0.109711069
      CC(2,3,1)= 6.654801E-02
      CC(3,1,1)=-0.116380157
      CC(3,2,1)= 0.217643894
      CC(3,3,1)= -9.767068E-02
```

```
      CT(1,1)= 0.834
      CT(2,1)= 0.977
      IF (IBOTH-1) 33,99,33
   33 CC(1,1,IBOTH)= 3.53503924
      CC(1,2,IBOTH)=-3.34778216
      CC(1,3,IBOTH)= 2.87262413
      CC(2,1,IBOTH)=11.2616503
      CC(2,2,IBOTH)=-27.4162512
      CC(2,3,IBOTH)= 21.7308359
      CC(3,1,IBOTH)=-23.7915472
      CC(3,2,IBOTH)= 44.2607764
      CC(3,3,IBOTH)=-14.4996046
      CT(1,IBOTH)= 0.622
      CT(2,IBOTH)= 0.885
      GO TO 51
   34 CC(1,1,1)=  0.104115
      CC(1,2,1)= -0.230347
      CC(1,3,1)=  0.167644
      CC(2,1,1)= -0.194037
      CC(2,2,1)=  0.401478
      CC(2,3,1)= -0.164612
      CC(3,1,1)=  7.33246E-02
      CC(3,2,1)= -2.03275E-02
      CC(3,3,1)=  2.44682E-03
      CT(1,1)= 1.032
      CT(2,1)= 1.30
  99  DO 100 I=1,3
      DO 100 J=1,3
      CC(I,J,2)=0.0
  100 CONTINUE
      CT(1,2)=0.0
      CT(2,2)=0.0
   51 RETURN
      END
      SUBROUTINE TRAJ
      COMMON/AREA/ CT(2,2), CC(3,3,2), IREF,
     1CFORM1, DKG1, DMAX, ITYPE, IBOTH, DM1,CFORM2, DM2,
     2DKG2, DS, SL, FN, VMUZ, RAD, FRACT, VE, THETA, U, DEL,
     3V, VXA, VYA, CF, DM, DKG, VX, VY, TH, Y, YA,
     4VXO, YO, VYO, AN1, AN2, AP1, AP2, IREG, RHO, DTI, DTV
     5    ,G, A, AA,VYK,D,CKDG,MSTG
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
C INITIALIZE THE VARIABLES FOR THE TRAJECTORY SUBROUTINE
      CF= CFORM1
      DM= DM1
      DKG= DKG1
      MSTG= 1
      X= 0.0
      T= 0.0
      VX= VXA
      VY= VYA
      TH= FN
      Y= ALT
      YA= Y
C TYPE OF DRAG
      IF (ITYPE) 2,1,1
C SET STEP SIZE FOR FIRST STAGE DRAG
    1 D= DS+SL*U
      GO TO 3
C COMPUTE STEP SIZE
    2 D = DMAX
C CALL RUNGE KUTTA SUBROUTINE
    3 CALL RUNGE
      DTV = 1/G*(VY+SQRT(VY**2+2.*G*(Y)))
      D = DTI
      IF ((IDNO.LE.17).OR.(IDNO.EQ.23)) GO TO 4
C SET THE SECOND STAGE DRAG PARAMETERS
      MSTG = 2
      IF (ITYPE.EQ.2) MSTG = 1
      DKG= DKG2
      DM= DM2
```

```
      CF = CFORM2
      TH = 0.0
    4 IF (DTV - D) 5,3,3
C SET THE STEP SIZE TO THE VACUUM DROP TIME REMAINING
    5 D= DTV
C SET THE DRAG PARAMETERS FOR THE FINAL INTEGRATICN STEP
      MSTG= 2
      IF (ITYPE.EQ.2) MSTG = 1
      DKG= DKG2
      DM= DM2
      TH=0.0
      CF= CFORM2
C CALL RUNGE FOR THE FINAL INTEGRATION STEP
      CALL RUNGE
      DTV = 1/G*(VY+SQRT(VY**2+2.*G*(Y)))
C UPDATE THE TIME OF FALL AND THE DOWN RANGE TRAVEL
      T = T + DTV
      X = X +DTV*VX
      RETURN
      END
      SUBROUTINE RUNGE
      COMMON/AREA/ CT(2,2), CC(3,3,2), IREF,
     1CFORM1, DKG1, DMAX, ITYPE, IBOTH, DM1,CFORM2, DM2,
     2DKG2, DS, SL, FN, VMUZ, RAD, FRACT, VE, THETA, U, DEL,
     3V, VXA, VYA, CF, DM, DKG, VX, VY, TH, Y, YA,
     4VXO, YO, VYO, AN1, AN2, AP1, AP2, IREG, RHO, DTI, DTV
     5    ,G, A, AA,VYK,D,CKDG,MSTG
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
C INITIALIZE THE VARIABLES FOR THE RUNGE KUTTA
      AD = A*D
      YO= Y
      VXO= VX
      VYO= VY
      RHO= 2.37576E-03-Y*(6.87557E-08-Y*6.71618E-13)
      CALL DERIV
C UPDATE POSITION AND VELOCITIES
      Y= YO+AD*VY
      RHO= 2.37576E-03-Y*(6.87557E-08-Y*6.71618E-13)
      AP1= AP2
      AN1= AN2
      VX= VXO+AD*AN1
      VY= VYO+AD*AP1
      CALL DERIV
C COMPUTE TIME, POSITION AND VELOCITIES
      T= T+D
      X= X+D*(VXO+AA*(VX-VXO))
      Y= YO+D*(VYO+AA*(VY-VYO))
      VX= VXO+D*(AN1+AA*(AN2-AN1))
      VY= VYO+D*(AP1+AA*(AP2-AP1))
      RETURN
      END
      SUBROUTINE DERIV
      COMMON/AREA/ CT(2,2), CC(3,3,2), IREF,
     1CFORM1, DKG1, DMAX, ITYPE, IBOTH, DM1,CFORM2, DM2,
     2DKG2, DS, SL, FN, VMUZ, RAD, FRACT, VE, THETA, U, DEL,
     3V, VXA, VYA, CF, DM, DKG, VX, VY, TH, Y, YA,
     4VXO, YO, VYO, AN1, AN2, AP1, AP2, IREG, RHO, DTI, DTV
     5    ,G, A, AA,VYK,D,CKDG,MSTG
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
C COMPUTE THE TOTAL VELOCITY AND THE MACH OF THE WEAPON
      V = SQRT(VX*VX+VY*VY)
      CM= V*(8.9544E-04+3.26E-09*Y)+DM
C DETERMINE THE REGION OF THE DRAG CURVE THAT IS APPLICABLE
      IF (CM-CT(1,MSTG)) 1,1,2
    1 IREG= 1
      GO TO 5
    2 IF (CM-CT(2,MSTG)) 3,3,4
    3 IREG= 2
      GO TO 5
    4 IREG= 3
```

```
C DO INTERMECIATE BALLISTIC CALCULATION
    5 CKDG = DKG+CF*(CC(IREG,1,MSTG)+(CC(IREG,2,MSTG)
     1+CC(IREG,3,MSTG)*CM)*CM)
      HH= TH/V-RHO*CKDG*V
      AN2= HH*VX
      AP2= HH*VY-G
      RETURN
      END
      SUBROUTINE INTER(XARP,TARP)
C    ****************************************************
C    *                                                  *
C    *   SETS UP THE INCREMENTS OF TIME AND THE         *
C    *   DIRECTION OF SEARCH OF THE INTERSECTION.       *
C    *                                                  *
C    ****************************************************
      COMMCN/TIM/RTIME(4),DRO(4),DTT(4)
      CCMMON/RET/RT,ZERO
      RT=RTIME(1)
      DT=1.
C           RES=1.0 MEANS THE ANSWER WAS FOUND.
C    RES=0.0 MEANS: NO SOLUTICN.
      RES=COMP(DT)
      IF (RES .EQ. 0.0) GO TO 2
      IF (RES .EQ. 1.0) GO TO 1
      DT=-0.1
      RES=COMP(DT)
      IF (RES .EQ. 1.0) GO TO 1
      DT=0.01
      RES=CCMP(DT)
      IF (RES .EQ. 1.0) GO TO 1
      DT=-0.0009
      RES=COMP(DT)
    1 TARP=RT
      XARP=ZERO
      GO TO 3
    2 TARP=0.0
    3 RETURN
      END
      FUNCTICN COMP(DT)
C    ****************************************************
C    *                                                  *
C    *   EXECUTES THE SEARCH UNTIL THERE IS A CHANGE    *
C    *   OF SIGN OR THE INTERSECTION WAS FOUND.         *
C    *                                                  *
C    ****************************************************
      CCMMON/TIM/RTIME(4),DRO(4),DTT(4)
      COMMCN/RET/RT,ZERO
C     ***"ZERC",FOR THE XAC ARRAY.***
C     ***"ONE",FOR THE XDROP ARRAY.***
      CCMP=-1.0
      DC 1 I=1,50
      ZERO=ORTFIT(RT,DTT)
      ONE=ORTFIT(RT,DRC)
      IF((CT .EQ. 1.) .AND. ((ZERC .LE. 0.0) .OR.
     1(ONE .LE. 0.0)))COMP=0.0
      IF (ABS(ONE-ZERO).LT.0.001) GO TO 2
      IF (DT .GT. 0.0) GO TC 3
      IF (ZERC.GT.ONE) RETURN
      GC TO 1
    3 IF (ONE.GT.ZERO) RETURN
    1 RT=RT+DT
      CCMP=0.0
C     COMP=0.0 MEANS: NO SOLUTION.
      GO TO 4
    2 CCMP=1.0
C         COMP=1.0 MEANS THE ANSWER WAS FOUND.
    4 RETURN
      END
      FUNCTION ORTFIT(RT,RARRAY)
C    ****************************************************
C    *                                                  *
C    *   THIS FUNCTION   GENERATES THE NECESSARY        *
```

```
C    *   ORTHOGONAL POLYNOMIALS ,    FINDS THE                        *
C    *   COEFFICIENTS OF THE POLYNOMIAL THAT                          *
C    *   REPRESENT THE DATA POINTS AND EXTRAPOLATE FOR                *
C    *   THE REQUIRED TIME.                                           *
C    *                                                                *
C    ******************************************************************
      COMMON/TIM/RTIME(4)
      DIMENSION RARRAY(4)
      DIMENSION ARRAY(4)
      DIMENSION C(4)
C    *** RT: REAL TIME
      DO 1 I=1,4
  1   ARRAY(I)=RARRAY(I)
      S0=4.0
      B=0.0
      DO 8 I=1,4
  8   B=B+RTIME(I)
      B0=B/S0
      S1=0.0
      DO 9 I=1,4
  9   S1=S1+(RTIME(I)-B0)**2
      B=0.0
      DO 10 I=1,4
 10   B=B+(RTIME(I))*(RTIME(I)-B0)**2
      B1=B/S1
      C1=S1/S0
      S2=0.0
      DO 11 I=1,4
 11   S2=S2+((RTIME(I)-B1)*(RTIME(I)-B0)-C1)**2
      B=0.0
      DO 111 I=1,4
111   B=B+RTIME(I)*((RTIME(I)-B1)*(RTIME(I)-B0)-C1)**2
      B2=B/S2
      C2=S2/S1
      S3=0.0
      DO 112 I=1,4
112   S3=S3+((RTIME(I)-B2)*((RTIME(I)-B1)*(RTIME(I)-B0)-C1)-
     1C2*(RTIME(I)-B0))**2
      D=0.0
      DO 12 I=1,4
 12   D=D+ARRAY(I)
      C(1)=D/S0
      D=0.0
      DO 13 I=1,4
 13   D=D+(ARRAY(I))*(RTIME(I)-B0)
      C(2)=D/S1
      D=0.0
      DO 14 I=1,4
 14   D=D+((ARRAY(I))*((RTIME(I)-B1)*(RTIME(I)-B0)-C1))
      C(3)=D/S2
      D=0.0
      DO 141 I=1,4
141   D=D+(ARRAY(I))*((RTIME(I)-B2)*((RTIME(I)-B1)*(RTIME(I)
     1-B0)-C1)-C2*(RTIME(I)-B0))
      C(4)=D/S3
      ORTFIT=C(1)+C(2)*(RT-B0)+C(3)*((RT-B1)*(RT-B0)-C1)
     1 +C(4)*((RT-B2)*((RT-B1)*(RT-B0)-C1)-C2*(RT-B0))
      RETURN
      END
```

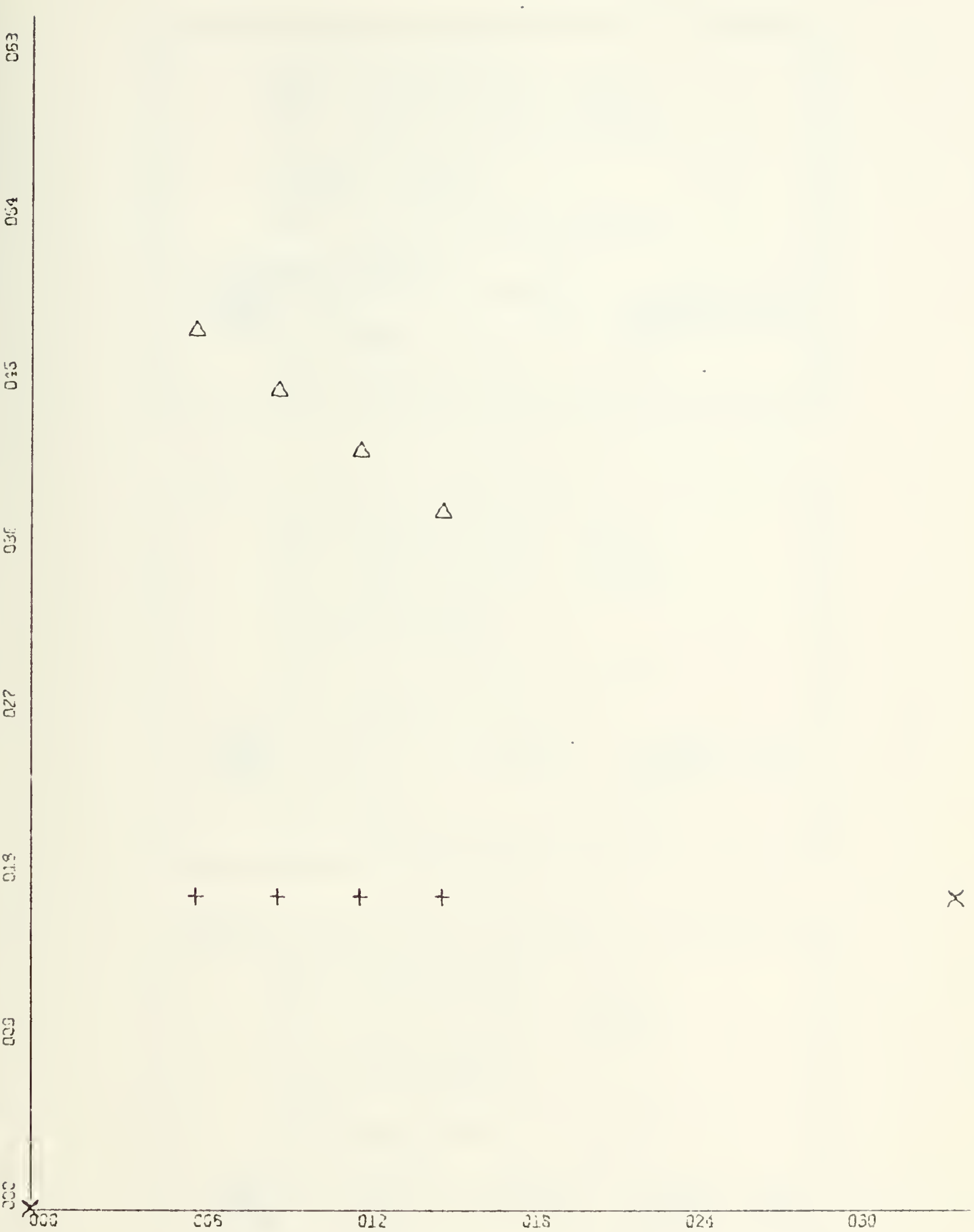
```
RUN NUMBER 1  IDNO=10


DATA:

TIME  VKTS   DEG     ALT     DTT     DRT
 0.0   0.0   0.0  7549.4  4148.9     0.0
 1.0 398.0 -30.5  7207.8  3958.5  2468.3
 2.0 406.8 -29.9  6865.5  3762.6  2446.4
 3.0 415.2 -29.3  6522.3  3561.4  2417.6
 4.0 423.4 -28.8  6178.3  3355.3  2376.7
 5.0 431.1 -28.3  5833.3  3144.3  2328.3
 6.0 438.5 -27.8  5487.4  2928.9  2273.4
 7.0 445.5 -27.5  5140.5  2709.3  2198.8
 8.0 452.1 -27.1  4792.5  2485.8  2122.3
 9.0 458.4 -26.8  4443.4  2258.5  2032.5
 9.5 462.8 -26.6  4268.4  2143.5  1990.0
 9.6 464.5 -26.5  4233.4  2120.4  1986.2
 9.7 465.1 -26.5  4198.4  2097.3  1974.6
 9.8 465.6 -26.5  4163.3  2074.2  1962.7
 9.9 466.2 -26.4  4128.3  2051.0  1956.5
10.0 466.7 -26.4  4093.2  2027.8  1944.5
10.1 467.3 -26.4  4058.1  2004.5  1932.6
10.2 467.8 -26.4  4023.0  1981.2  1920.5
10.3 468.3 -26.4  3987.9  1957.9  1908.3
10.4 468.9 -26.3  3952.8  1934.6  1901.6
10.5 469.4 -26.3  3917.6  1911.2  1889.3
10.6 469.9 -26.3  3882.5  1887.8  1876.9
10.7 470.4 -26.3  3847.3  1864.3  1864.4
10.8 471.0 -26.2  3812.1  1840.9  1857.3
10.9 471.5 -26.2  3776.9  1817.4  1845.4
11.0 472.0 -26.2  3741.7  1793.9  1832.7
11.1 472.5 -26.2  3706.5  1770.3  1819.9
11.2 473.0 -26.2  3671.3  1746.7  1807.0
11.3 473.5 -26.1  3636.1  1723.1  1799.4
11.4 474.0 -26.1  3600.8  1699.5  1786.3
11.5 474.5 -26.1  3565.5  1675.8  1773.2
11.6 474.9 -26.1  3530.3  1652.1  1759.9
11.7 475.4 -26.1  3495.0  1628.4  1746.6
11.8 475.9 -26.1  3459.7  1604.6  1733.3
11.9 476.4 -26.0  3424.3  1580.8  1725.0
12.0 476.9 -26.0  3389.0  1557.0  1711.5
12.1 477.3 -26.0  3353.7  1533.2  1697.8
12.2 477.8 -26.0  3318.3  1509.3  1684.2
12.3 478.2 -26.0  3282.9  1485.5  1670.3
12.4 478.7 -26.0  3247.6  1461.5  1656.6

EXTRAPOLATION AT TIME 10.0:
VKTS=463.9 DEG=-26.7 ALT=4093.5 DTT=2027.8 DRT=1925.5


EXTRAPOLATION AT TIME 11.0:
VKTS=469.0 DEG=-26.5 ALT=3742.3 DTT=1794.0 DRT=1812.6


EXTRAPOLATION AT TIME 12.0:
VKTS=473.6 DEG=-26.4 ALT=3389.9 DTT=1557.3 DRT=1688.4


DOWNRANGE TRAVEL FOUND BY EXTRAPOLATION= 1831.4
TIME OF RELEASE FOUND BY EXTRAPOLATION=10.84
```

```
RUN NUMBER 2  IDNO=10


DATA:

TIME  VKTS    DEG     ALT     DTT     DRT
 0.0   0.0    0.0  9225.4  3896.0     0.0
 1.0 343.6  -37.0  8876.3  3743.5  2269.0
 2.0 355.2  -37.2  8513.7  3586.3  2235.0
 3.0 366.5  -37.2  8139.5  3424.1  2202.1
 4.0 377.5  -37.1  7755.3  3256.7  2164.9
 5.0 388.1  -36.8  7362.9  3084.0  2129.0
 6.0 398.4  -36.4  6964.1  2905.8  2089.8
 7.0 408.4  -35.8  6560.6  2721.8  2052.4
 8.0 418.1  -35.1  6154.1  2531.9  2012.3
 9.0 427.6  -34.4  5746.5  2335.8  1963.5
 9.5 434.6  -33.7  5542.7  2235.4  1958.5
 9.6 437.4  -33.5  5502.0  2215.1  1963.1
 9.7 438.3  -33.4  5461.3  2194.8  1958.6
 9.8 439.2  -33.3  5420.6  2174.4  1954.2
 9.9 440.1  -33.2  5379.9  2153.9  1949.6
10.0 441.0  -33.1  5339.3  2133.4  1944.9
10.1 441.9  -33.0  5298.7  2112.8  1940.2
10.2 442.8  -32.9  5258.1  2092.1  1935.4
10.3 443.8  -32.8  5217.5  2071.4  1930.7
10.4 444.7  -32.7  5177.0  2050.6  1925.8
10.5 445.6  -32.6  5136.5  2029.8  1920.7
10.6 446.5  -32.5  5096.0  2008.8  1915.6
10.7 447.4  -32.4  5055.5  1987.8  1910.4
10.8 448.3  -32.3  5015.1  1966.8  1905.1
10.9 449.2  -32.2  4974.7  1945.6  1899.8
11.0 450.1  -32.0  4934.4  1924.5  1899.5
11.1 451.0  -31.9  4894.1  1903.2  1893.9
11.2 451.9  -31.8  4853.8  1881.9  1888.3
11.3 452.8  -31.7  4813.6  1860.5  1882.7
11.4 453.7  -31.6  4773.5  1839.0  1876.9
11.5 454.6  -31.5  4733.3  1817.5  1871.1
11.6 455.5  -31.4  4693.3  1795.9  1865.2
11.7 456.4  -31.3  4653.2  1774.2  1859.2
11.8 457.3  -31.2  4613.3  1752.5  1853.1
11.9 458.2  -31.1  4573.4  1730.7  1847.0
12.0 459.1  -30.9  4533.5  1708.8  1845.8
12.1 460.0  -30.8  4493.7  1686.8  1839.5
12.2 460.9  -30.7  4453.9  1664.8  1833.1
12.3 461.8  -30.6  4414.3  1642.7  1826.6
12.4 462.7  -30.5  4374.6  1620.6  1820.0

EXTRAPOLATION AT TIME 10.0:
VKTS=436.4 DEG=-33.2 ALT=5339.8 DTT=2134.0 DRT=1937.3


EXTRAPOLATION AT TIME 11.0:
VKTS=445.1 DEG=-32.0 ALT=4935.3 DTT=1925.5 DRT=1900.3


EXTRAPOLATION AT TIME 12.0:
VKTS=453.5 DEG=-30.7 ALT=4535.0 DTT=1710.4 DRT=1864.6


DOWNRANGE TRAVEL FOUND BY EXTRAPOLATION= 1895.2
TIME OF RELEASE FOUND BY EXTRAPOLATION=11.14
```

```
RUN NUMBER 3  IDNO=10


DATA:

TIME  VKTS    DEG    ALT     DTT     DRT
 0.0    0.0   0.0 2000.0 3699.4     0.0
 1.0 300.0    0.0 2000.0 3532.7  1837.2
 2.0 300.0    0.0 2000.0 3366.0  1837.2
 3.0 300.0    0.0 2000.0 3199.4  1837.2
 4.0 300.0    0.0 2000.0 3032.7  1837.2
 5.0 300.0    0.0 2000.0 2866.0  1837.2
 6.0 300.0    0.0 2000.0 2699.4  1837.2
 7.0 300.0    0.0 2000.0 2532.7  1837.2
 8.0 300.0    0.0 2000.0 2366.0  1837.2
 9.0 300.0    0.0 2000.0 2199.4  1837.2
 9.5 300.0    0.0 2000.0 2116.0  1837.2
 9.6 300.0    0.0 2000.0 2099.4  1837.2
 9.7 300.0    0.0 2000.0 2082.7  1837.2
 9.8 300.0    0.0 2000.0 2066.0  1837.2
 9.9 300.0    0.0 2000.0 2049.4  1837.2
10.0 300.0    0.0 2000.0 2032.7  1837.2
10.1 300.0    0.0 2000.0 2016.0  1837.2
10.2 300.0    0.0 2000.0 1999.4  1837.2
10.3 300.0    0.0 2000.0 1982.7  1837.2
10.4 300.0    0.0 2000.0 1966.0  1837.2
10.5 300.0    0.0 2000.0 1949.4  1837.2
10.6 300.0    0.0 2000.0 1932.7  1837.2
10.7 300.0    0.0 2000.0 1916.0  1837.2
10.8 300.0    0.0 2000.0 1899.4  1837.2
10.9 300.0    0.0 2000.0 1882.7  1837.2
11.0 300.0    0.0 2000.0 1866.0  1837.2
11.1 300.0    0.0 2000.0 1849.4  1837.2
11.2 300.0    0.0 2000.0 1832.7  1837.2
11.3 300.0    0.0 2000.0 1816.0  1837.2
11.4 300.0    0.0 2000.0 1799.4  1837.2
11.5 300.0    0.0 2000.0 1782.7  1837.2
11.6 300.0    0.0 2000.0 1766.0  1837.2
11.7 300.0    0.0 2000.0 1749.4  1837.2
11.8 300.0    0.0 2000.0 1732.7  1837.2
11.9 300.0    0.0 2000.0 1716.0  1837.2
12.0 300.0    0.0 2000.0 1699.4  1837.2
12.1 300.0    0.0 2000.0 1699.4  1837.2
12.2 300.0    0.0 2000.0 1666.0  1837.2
12.3 300.0    0.0 2000.0 1649.4  1837.2
12.4 300.0    0.0 2000.0 1632.7  1837.2

EXTRAPOLATION AT TIME 10.0:
VKTS=300.0 DEG=  0.0 ALT=2000.0 DTT=2033.4 DRT=1837.2


EXTRAPOLATION AT TIME 11.0:
VKTS=300.0 DEG=  0.0 ALT=2000.0 DTT=1867.2 DRT=1837,2


EXTRAPOLATION AT TIME 12.0:
VKTS=300.0 DEG=  0.0 ALT=2000.0 DTT=1701.2 DRT=1837.2


DOWNRANGE TRAVEL FOUND BY EXTRAPOLATION= 1837.3
TIME OF RELEASE FOUND BY EXTRAPOLATION=11.18
```

```
RUN NUMBER 4  IDNO=10


DATA:

TIME  VKTS    DEG     ALT    DTT     DRT
 0.0   0.0    0.0  4451.7 5228.2     0.0
 1.0 450.0  -10.0  4319.7 4982.0  3106.4
 2.0 450.0  -10.0  4187.7 4735.8  3047.7
 3.0 450.0  -10.0  4055.8 4489.6  2988.1
 4.0 450.0  -10.0  3923.8 4243.4  2927.6
 5.0 450.0  -10.0  3791.8 3997.2  2866.1
 6.0 450.0  -10.0  3659.8 3751.0  2803.6
 7.0 450.0  -10.0  3527.9 3504.8  2740.0
 8.0 450.0  -10.0  3395.9 3258.6  2675.2
 9.0 450.0  -10.0  3263.9 3012.4  2610.1
 9.5 450.0  -10.0  3198.0 2889.3  2576.6
 9.6 450.0  -10.0  3184.8 2864.7  2569.8
 9.7 450.0  -10.0  3171.6 2840.1  2563.1
 9.8 450.0  -10.0  3158.4 2815.4  2556.3
 9.9 450.0  -10.0  3145.2 2790.8  2549.5
10.0 450.0  -10.0  3132.0 2766.2  2542.7
10.1 450.0  -10.0  3118.8 2741.6  2535.9
10.2 450.0  -10.0  3105.6 2717.0  2529.1
10.3 450.0  -10.0  3092.4 2692.3  2522.2
10.4 450.0  -10.0  3079.2 2667.7  2515.4
10.5 450.0  -10.0  3066.0 2643.1  2508.5
10.6 450.0  -10.0  3052.8 2618.5  2501.6
10.7 450.0  -10.0  3039.6 2593.9  2494.7
10.8 450.0  -10.0  3026.4 2569.2  2487.8
10.9 450.0  -10.0  3013.2 2544.6  2480.9
11.0 450.0  -10.0  3000.0 2520.0  2474.0
11.1 450.0  -10.0  2986.8 2495.4  2467.0
11.2 450.0  -10.0  2973.6 2470.8  2460.0
11.3 450.0  -10.0  2960.4 2446.1  2453.1
11.4 450.0  -10.0  2947.2 2421.5  2446.1
11.5 450.0  -10.0  2934.0 2396.9  2439.1
11.6 450.0  -10.0  2920.8 2372.3  2432.0
11.7 450.0  -10.0  2907.6 2347.7  2425.0
11.8 450.0  -10.0  2894.4 2323.0  2417.9
11.9 450.0  -10.0  2881.2 2298.4  2410.9
12.0 450.0  -10.0  2868.0 2273.8  2403.8
12.1 450.0  -10.0  2854.8 2249.2  2396.7
12.2 450.0  -10.0  2841.6 2224.6  2389.6
12.3 450.0  -10.0  2828.4 2199.9  2382.5
12.4 450.0  -10.0  2815.2 2175.3  2375.3

EXTRAPOLATION AT TIME 10.0:
VKTS=450.0 DEG=-10.0 ALT=3132.7 DTT=2766.2 DRT=2542.5


EXTRAPOLATION AT TIME 11.0:
VKTS=450.0 DEG=-10.0 ALT=3001.2 DTT=2520.0 DRT=2474.3


EXTRAPOLATION AT TIME 12.0:
VKTS=450.0 DEG=-10.0 ALT=2869.9 DTT=2273.8 DRT=2404.9


DOWNRANGE TRAVEL FOUND BY EXTRAPOLATION= 2456.6
TIME OF RELEASE FOUND BY EXTRAPOLATION=11.26
```

```
****************************************
*                                      *
*              INPUT DATA:             *
*    VKTS=  400.00    DTNAV=   0.200   *
*     XAT=    5.00    DTBAL=   3.000   *
*     ALT=  4000.0     DTE1=   0.002   *
*     DEG=     5.0     DTE2=   0.002   *
*    TIME=     0.0     DTE3=   0.300   *
*      YT=     0.0     IDNO=      11   *
*      TC=  18.313      TMV=   3.000   *
*    ATOL=     5.0     RLIM=     0.5   *
*    ALIM=   200.0     DTE4=   0.004   *
*    DTE5=   0.005     DTE6=   0.005   *
*                                      *
****************************************
****************************************
*                                      *
*      PRELIMINARY  COMPUTATIONS:      *
*                                      *
*      AT TIME= 6.004 :                *
*      XAT=  4.35   ALT=  4346.1       *
*                                      *
*      THE TIME NOW IS:   6.006        *
*                                      *
*      AT TIME= 6.004 :                *
*      XDROP=  1.90                    *
*                                      *
*      THE TIME NOW IS:   9.008        *
*                                      *
*      AT TIME= 9.008 :                *
*      XAT=  4.03   ALT=  4513.6       *
*                                      *
*      THE TIME NOW IS:   9.010        *
*                                      *
*      AT TIME= 9.008 :                *
*      XDROP=  1.89                    *
*                                      *
*      THE TIME NOW IS:  12.012        *
*                                      *
*      AT TIME=12.012 :                *
*      XAT=  3.73   ALT=  4677.5       *
*                                      *
*      THE TIME NOW IS:  12.014        *
*                                      *
*      AT TIME=12.012 :                *
*      XDROP=  1.87                    *
*                                      *
*      THE TIME NOW IS:  15.016        *
*                                      *
*      AT TIME=15.016 :                *
*      XAT=  3.43   ALT=  4837.6       *
*                                      *
*      THE TIME NOW IS:  15.018        *
*                                      *
*      AT TIME=15.016 :                *
*      XDROP=  1.85                    *
*                                      *
*      THE TIME NOW IS:  18.020        *
*                                      *
*                                      *
****************************************
****************************************
*                                      *
*  RELEASE POINT:                      *
*  HORIZ. DIST. TO TARGET:   1.685  NM *
*  TIME                  :  33.904 SEC *
*  A/C ALTITUDE          :  5758.9  FT *
*                                      *
*      THE TIME NOW IS:  18.325        *
*                                      *
****************************************
```



X-SCALE=6.00E+00 UNITS INCH.
Y-SCALE=9.00E-01 UNITS INCH.
CURVES OF XAC AND XDROP,VS. TIME

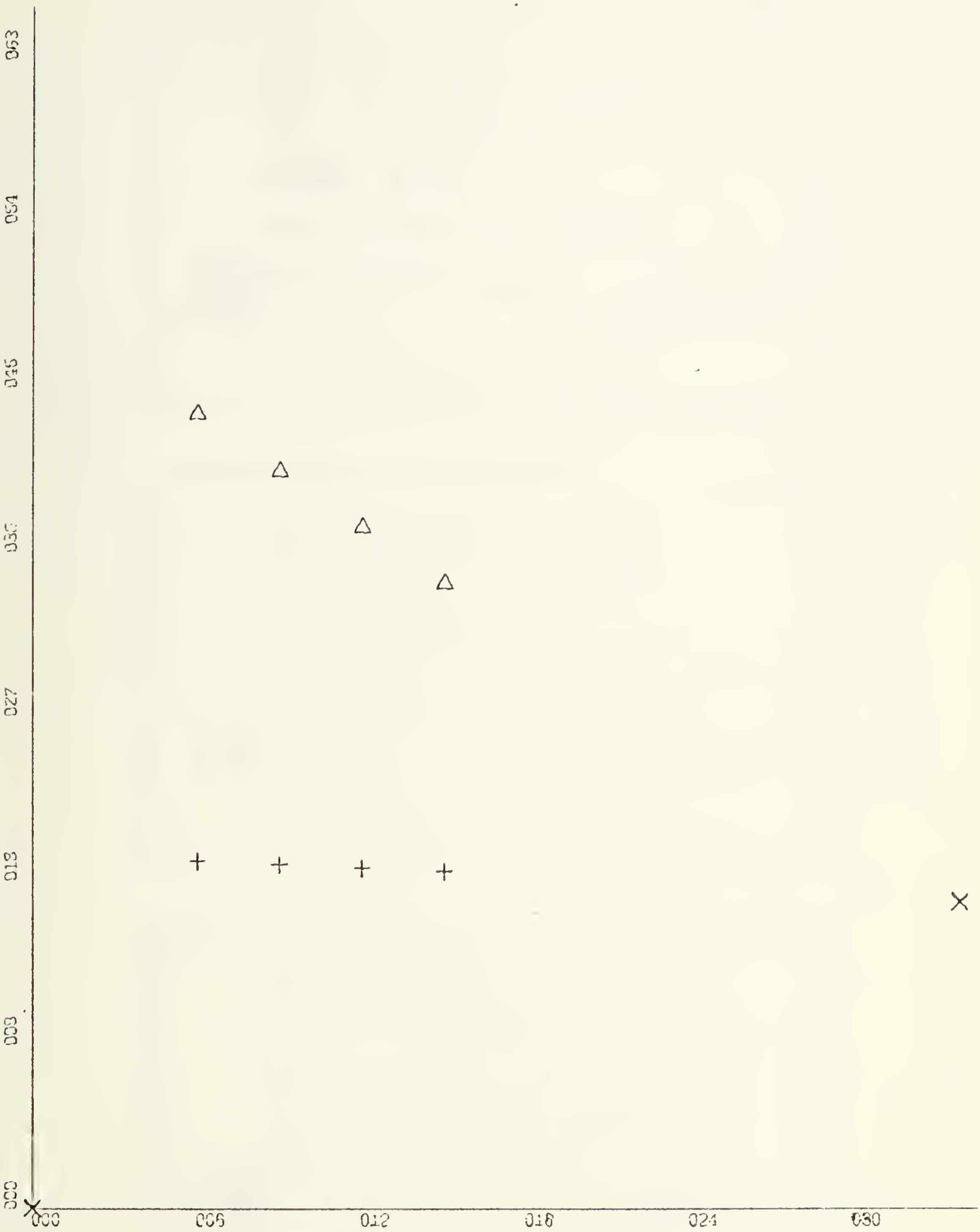
```
****************************************
*                                      *
*      AT TIME=18.325 :                *
*      XAT=  3.10  ALT=  5009.6        *
*                                      *
*      THE TIME NOW IS:  18.330        *
*                                      *
*      AT TIME=18.325 :                *
*      XDROP=  1.83                    *
*                                      *
*      THE TIME NOW IS:  21.332        *
*                                      *
*  RELEASE POINT:                      *
*  HORIZ. DIST. TO TARGET:   1.690  NM *
*  TIME                  :  33.908 SEC *
*  A/C ALTITUDE          :  5759.1  FT *
*                                      *
*      THE TIME NOW IS:  21.832        *
*                                      *
****************************************

****************************************
*                                      *
*      AT TIME=21.832 :                *
*      XAT=  2.77  ALT=  5187.0        *
*                                      *
*      THE TIME NOW IS:  21.837        *
*                                      *
*      AT TIME=21.832 :                *
*      XDROP=  1.80                    *
*                                      *
*      THE TIME NOW IS:  24.839        *
*                                      *
*  RELEASE POINT:                      *
*  HORIZ. DIST. TO TARGET:   1.689  NM *
*  TIME                  :  33.952 SEC *
*  A/C ALTITUDE          :  5761.1  FT *
*                                      *
*      THE TIME NOW IS:  25.339        *
*                                      *
****************************************

****************************************
*                                      *
*      AT TIME=25.339 :                *
*      XAT=  2.44  ALT=  5359.4        *
*                                      *
*      THE TIME NOW IS:  25.344        *
*                                      *
*      AT TIME=25.339 :                *
*      XDROP=  1.77                    *
*                                      *
*      THE TIME NOW IS:  28.346        *
*                                      *
*  RELEASE POINT:                      *
*  HORIZ. DIST. TO TARGET:   1.689  NM *
*  TIME                  :  33.956 SEC *
*  A/C ALTITUDE          :  5761.3  FT *
*                                      *
*      THE TIME NOW IS:  28.846        *
*                                      *
****************************************
```

```
******************************************
*                                        *
*       AT TIME=28.846 :                 *
*       XAT=  2.13  ALT=  5526.6         *
*                                        *
*       THE TIME NOW IS:  28.851         *
*                                        *
*       AT TIME=28.846 :                 *
*       XDROP=  1.74                     *
*                                        *
*       THE TIME NOW IS:  31.853         *
*                                        *
*   RELEASE POINT:                       *
*   HORIZ. DIST. TO TARGET:   1.689  NM  *
*   TIME                  :  33.955 SEC  *
*   A/C ALTITUDE          :  5761.2  FT  *
*                                        *
*       THE TIME NOW IS:  32.353         *
*                                        *
******************************************
```

```
******************************************
*                                        *
*       NOT ENOUGH TIME TO COMPUTE.      *
*                                        *
******************************************
```

```
*****************************************
*                                       *
*            INPUT DATA:                *
*   VKTS=  400.00    DTNAV=   0.200     *
*    XAT=    1.90    DTBAL=   3.000     *
*    ALT=  4000.0     DTE1=   0.002     *
*    DEG=     5.0     DTE2=   0.002     *
*   TIME=     0.0     DTE3=   0.300     *
*     YT=     0.0     IDNO=      11     *
*     TC=  18.313      TMV=   3.000     *
*   ATOL=     5.0     RLIM=     0.5     *
*   ALIM=   200.0     DTE4=   0.004     *
*   DTE5=   0.005     DTE6=   0.005     *
*                                       *
*****************************************

*****************************************
*                                       *
*     NO SOLUTION FOR THESE DATA:       *
*                                       *
*****************************************
```

```
*****************************************
*                                       *
*              INPUT DATA:              *
*    VKTS=  400.00    DTNAV=   0.200    *
*     XAT=    5.50    DTBAL=   3.000    *
*     ALT=  4000.0     DTE1=   0.002    *
*     DEG=     0.0     DTE2=   0.002    *
*    TIME=     0.0     DTE3=   0.300    *
*      YT=     0.0     IDNO=      11    *
*      TC=  18.313      TMV=   3.000    *
*    ATOL=     5.0     RLIM=     0.5    *
*    ALIM=   200.0     DTE4=   0.004    *
*    DTE5=   0.005     DTE6=   0.005    *
*                                       *
*****************************************
*****************************************
*                                       *
*      PRELIMINARY  COMPUTATIONS:       *
*                                       *
*      AT TIME= 6.004 :                 *
*      XAT=  4.83  ALT=  4000.0         *
*                                       *
*      THE TIME NOW IS:   6.006         *
*                                       *
*      AT TIME= 6.004 :                 *
*      XDROP=  1.72                     *
*                                       *
*      THE TIME NOW IS:   9.008         *
*                                       *
*      AT TIME= 9.008 :                 *
*      XAT=  4.50  ALT=  4000.0         *
*                                       *
*      THE TIME NOW IS:   9.010         *
*                                       *
*      AT TIME= 9.008 :                 *
*      XDROP=  1.72                     *
*                                       *
*      THE TIME NOW IS:  12.012         *
*                                       *
*      AT TIME=12.012 :                 *
*      XAT=  4.17  ALT=  4000.0         *
*                                       *
*      THE TIME NOW IS:  12.014         *
*                                       *
*      AT TIME=12.012 :                 *
*      XDROP=  1.72                     *
*                                       *
*      THE TIME NOW IS:  15.016         *
*                                       *
*      AT TIME=15.016 :                 *
*      XAT=  3.83  ALT=  4000.0         *
*                                       *
*      THE TIME NOW IS:  15.018         *
*                                       *
*      AT TIME=15.016 :                 *
*      XDROP=  1.72                     *
*                                       *
*      THE TIME NOW IS:  18.020         *
*                                       *
*                                       *
*****************************************
*****************************************
*                                       *
*  RELEASE POINT:                       *
*  HORIZ. DIST. TO TARGET:   1.720  NM  *
*  TIME                  :  33.944 SEC  *
*  A/C ALTITUDE          :  4000.0  FT  *
*                                       *
*      THE TIME NOW IS:  18.325         *
*                                       *
*****************************************
```



X-SCALE=6.00E+00 UNITS INCH.
Y-SCALE=9.00E-01 UNITS INCH.

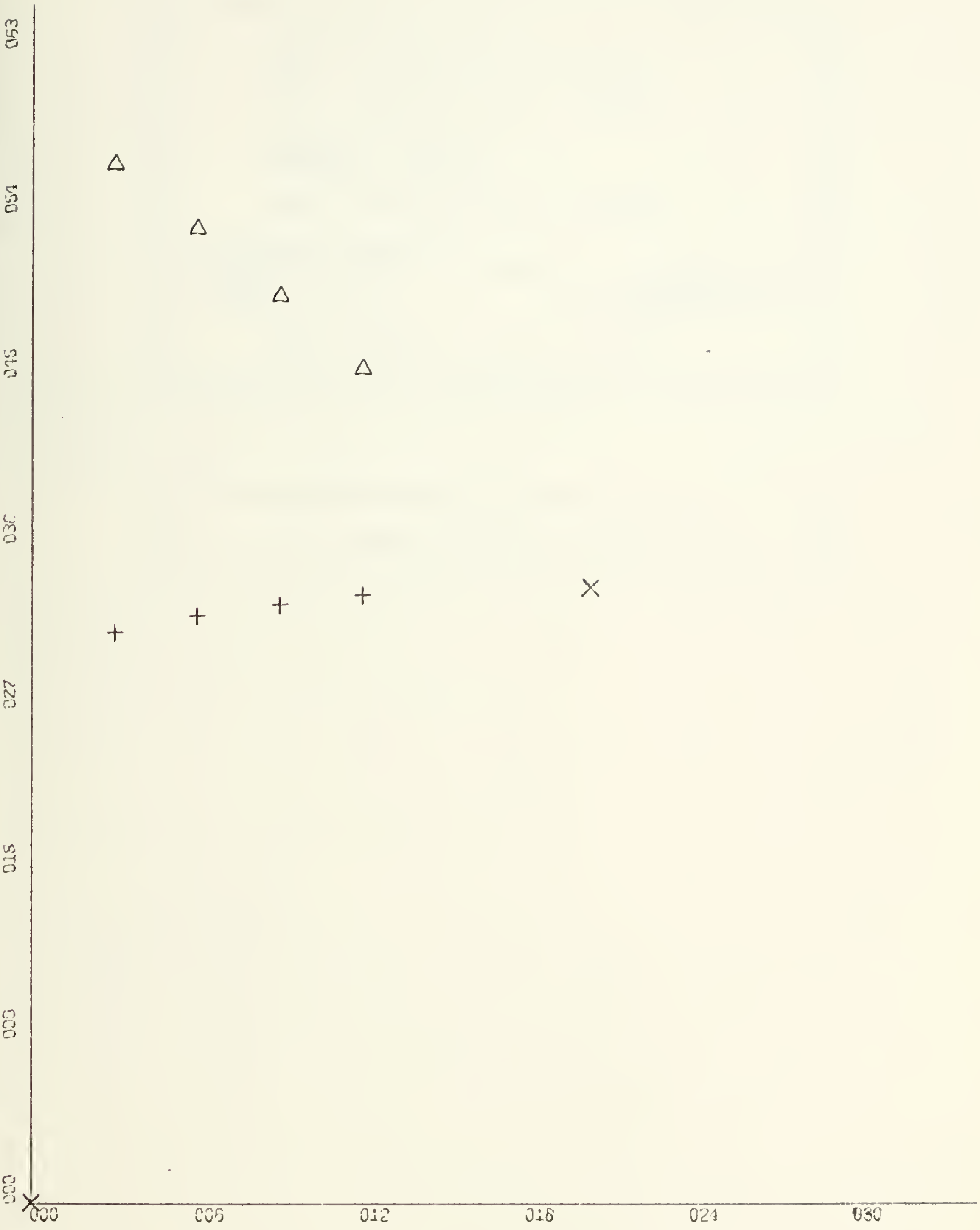
CURVES OF XAC AND XDROP, VS. TIME

```
******************************************
*                                        *
*      AT TIME=18.325 :                  *
*      XAT=  3.46  ALT=  4000.0          *
*                                        *
*      THE TIME NOW IS:  18.330          *
*                                        *
*      AT TIME=18.325 :                  *
*      XDROP=  1.72                      *
*                                        *
*      THE TIME NOW IS:  21.332          *
*                                        *
*  RELEASE POINT:                        *
*  HORIZ. DIST. TO TARGET:   1.721  NM   *
*  TIME                   :  33.978 SEC  *
*  A/C ALTITUDE           :  4000.0  FT  *
*                                        *
*      THE TIME NOW IS:  21.832          *
*                                        *
******************************************

******************************************
*                                        *
*      AT TIME=21.832 :                  *
*      XAT=  3.07  ALT=  4000.0          *
*                                        *
*      THE TIME NOW IS:  21.837          *
*                                        *
*      AT TIME=21.832 :                  *
*      XDROP=  1.72                      *
*                                        *
*      THE TIME NOW IS:  24.839          *
*                                        *
*  RELEASE POINT:                        *
*  HORIZ. DIST. TO TARGET:   1.723  NM   *
*  TIME                   :  33.992 SEC  *
*  A/C ALTITUDE           :  4000.0  FT  *
*                                        *
*      THE TIME NOW IS:  25.339          *
*                                        *
******************************************

******************************************
*                                        *
*      AT TIME=25.339 :                  *
*      XAT=  2.68  ALT=  4000.0          *
*                                        *
*      THE TIME NOW IS:  25.344          *
*                                        *
*      AT TIME=25.339 :                  *
*      XDROP=  1.72                      *
*                                        *
*      THE TIME NOW IS:  28.346          *
*                                        *
*  RELEASE POINT:                        *
*  HORIZ. DIST. TO TARGET:   1.722  NM   *
*  TIME                   :  33.996 SEC  *
*  A/C ALTITUDE           :  4000.0  FT  *
*                                        *
*      THE TIME NOW IS:  28.846          *
*                                        *
******************************************
```

```
*****************************************
*                                       *
*      AT TIME=28.846 :                 *
*      XAT=  2.29  ALT=  4000.0         *
*                                       *
*      THE TIME NOW IS:  28.851         *
*                                       *
*      AT TIME=28.846 :                 *
*      XDROP=  1.72                     *
*                                       *
*      THE TIME NOW IS:  31.853         *
*                                       *
*  RELEASE POINT:                       *
*  HORIZ. DIST. TO TARGET:   1.723  NM  *
*  TIME                  :  33.995 SEC  *
*  A/C ALTITUDE          :  4000.0  FT  *
*                                       *
*      THE TIME NOW IS:  32.353         *
*                                       *
*****************************************
```

```
*****************************************
*                                       *
*      NOT ENOUGH TIME TO COMPUTE.      *
*                                       *
*****************************************
```

```
*****************************************
*                                       *
*              INPUT DATA:              *
*    VKTS=  400.00    DTNAV=   0.200    *
*     XAT=    6.00    DTBAL=   3.000    *
*     ALT= 18240.0     DTE1=   0.002    *
*     DEG=   -15.0     DTE2=   0.002    *
*    TIME=     0.0     DTE3=   0.300    *
*      YT=     0.0     IDNO=      11    *
*      TC=  18.313      TMV=   3.000    *
*    ATOL=     5.0     RLIM=     0.5    *
*    ALIM=   200.0     DTE4=   0.004    *
*    DTE5=   0.005     DTE6=   0.005    *
*                                       *
*****************************************
*****************************************
*                                       *
*      PRELIMINARY  COMPUTATIONS:       *
*                                       *
*      AT TIME= 3.004 :                 *
*      XAT=  5.67  ALT= 17698.3         *
*                                       *
*      THE TIME NOW IS:   3.006         *
*                                       *
*      AT TIME= 3.004 :                 *
*      XDROP=  3.11                     *
*                                       *
*      THE TIME NOW IS:   6.008         *
*                                       *
*      AT TIME= 6.008 :                 *
*      XAT=  5.31  ALT= 17123.8         *
*                                       *
*      THE TIME NOW IS:   6.010         *
*                                       *
*      AT TIME= 6.008 :                 *
*      XDROP=  3.20                     *
*                                       *
*      THE TIME NOW IS:   9.012         *
*                                       *
*      AT TIME= 9.012 :                 *
*      XAT=  4.94  ALT= 16516.5         *
*                                       *
*      THE TIME NOW IS:   9.014         *
*                                       *
*      AT TIME= 9.012 :                 *
*      XDROP=  3.26                     *
*                                       *
*      THE TIME NOW IS:  12.016         *
*                                       *
*      AT TIME=12.016 :                 *
*      XAT=  4.55  ALT= 15876.3         *
*                                       *
*      THE TIME NOW IS:  12.018         *
*                                       *
*      AT TIME=12.016 :                 *
*      XDROP=  3.32                     *
*                                       *
*      THE TIME NOW IS:  15.020         *
*                                       *
*                                       *
*****************************************
*****************************************
*                                       *
*   RELEASE POINT:                      *
*   HORIZ. DIST. TO TARGET:   3.357  NM *
*   TIME                  :  20.314 SEC *
*   A/C ALTITUDE          : 13937.2  FT *
*                                       *
*      THE TIME NOW IS:  15.325         *
*                                       *
*****************************************
```

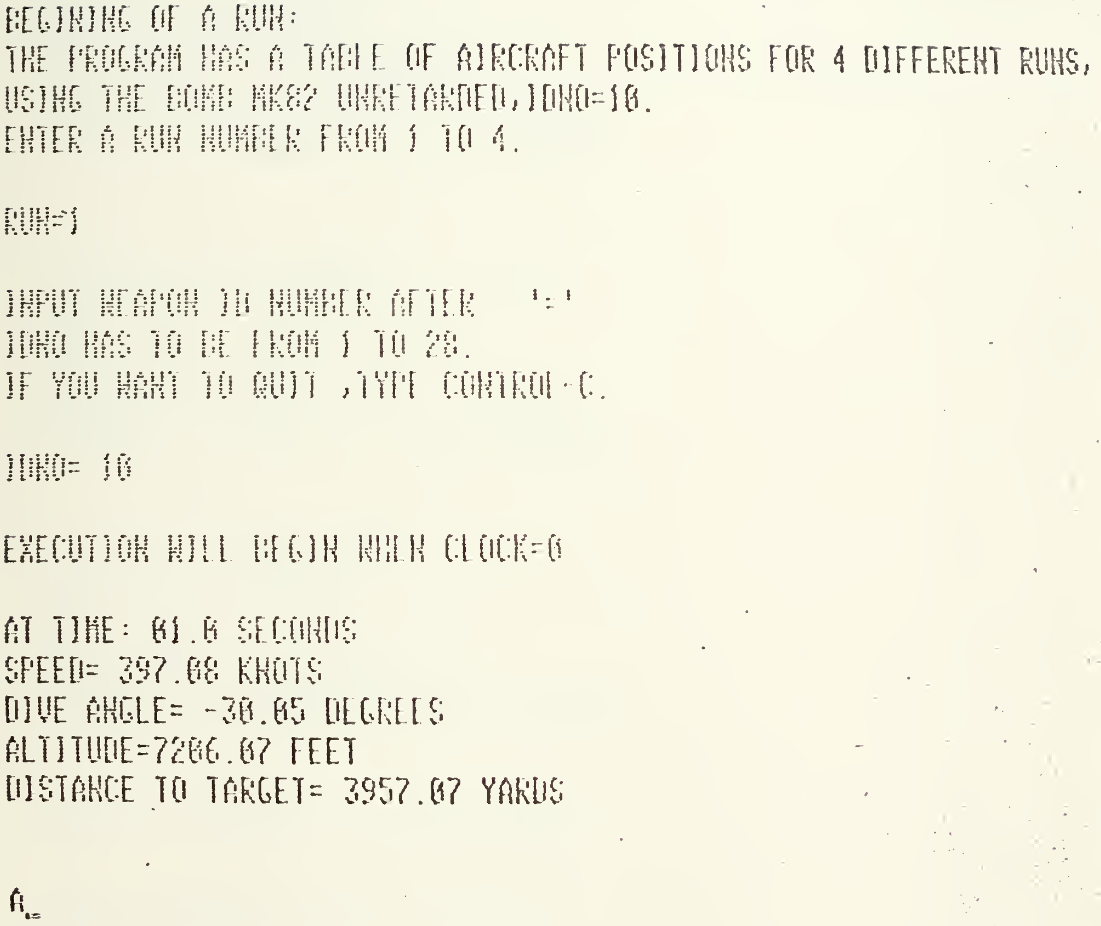


X-SCALE=6.00E+00 UNITS INCH.
Y-SCALE=9.00E-01 UNITS INCH.
CURVES OF XAC AND XDROP,VS. TIME

```
*****************************************
*                                       *
*      AT TIME=15.325 :                 *
*      XAT=  4.09  ALT= 15133.1         *
*                                       *
*      THE TIME NOW IS:  15.330         *
*                                       *
*      AT TIME=15.325 :                 *
*      XDROP=  3.35                     *
*                                       *
*      THE TIME NOW IS:  18.332         *
*                                       *
*  RELEASE POINT:                       *
*  HORIZ. DIST. TO TARGET:   3.353  NM  *
*  TIME                  :  20.358 SEC  *
*  A/C ALTITUDE          : 13926.3  FT  *
*                                       *
*      THE TIME NOW IS:  18.832         *
*                                       *
*****************************************
```

```
*****************************************
*                                       *
*      NOT ENOUGH TIME TO COMPUTE.      *
*                                       *
*****************************************
```

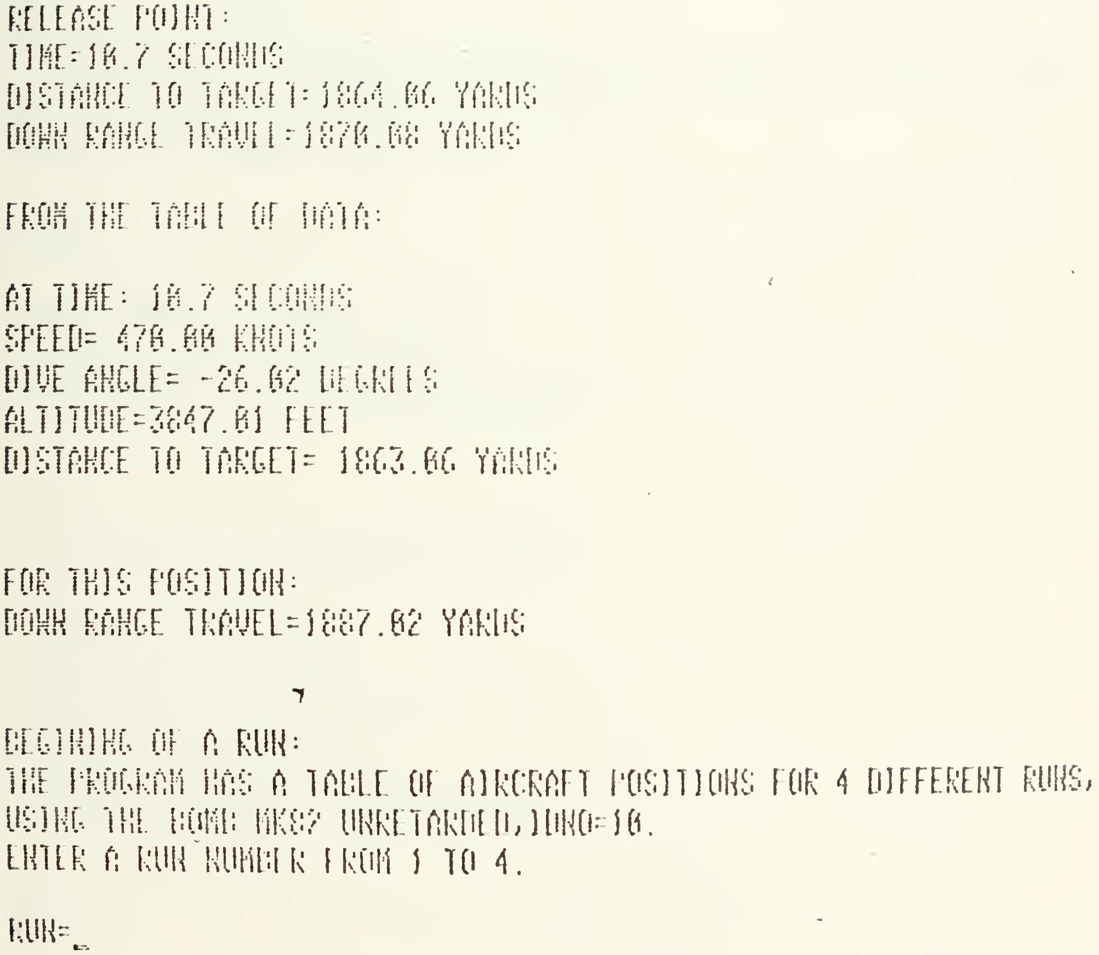
RUN NUMBER ONE:



```
BEGINING OF A RUN:
THE PROGRAM HAS A TABLE OF AIRCRAFT POSITIONS FOR 4 DIFFERENT RUNS,
USING THE BOMB MK82 UNRETARDED,IDNO=10.
ENTER A RUN NUMBER FROM 1 TO 4.

RUN=1

INPUT WEAPON ID NUMBER AFTER   '='
IDNO HAS TO BE FROM 1 TO 28.
IF YOU WANT TO QUIT ,TYPE CONTROL-C.

IDNO= 10

EXECUTION WILL BEGIN WHEN CLOCK=0

AT TIME: 01.0 SECONDS
SPEED= 397.08 KNOTS
DIVE ANGLE= -30.05 DEGREES
ALTITUDE=7206.07 FEET
DISTANCE TO TARGET= 3957.07 YARDS


A_
```

```
AT TIME: 63.0 SECONDS
SPEED= 415.02 KNOTS
DIVE ANGLE= -29.02 DEGREES
ALTITUDE=6521.08 FEET
DISTANCE TO TARGET= 3561.04 YARDS


AT TIME: 65.0 SECONDS
SPEED= 431.01 KNOTS
DIVE ANGLE= -28.07 DEGREES
ALTITUDE=5832.07 FEET
DISTANCE TO TARGET= 3143.06 YARDS


AT TIME: 67.0 SECONDS
SPEED= 445.04 KNOTS
DIVE ANGLE= -27.04 DEGREES
ALTITUDE=5140.03 FEET
DISTANCE TO TARGET= 2708.04 YARDS


START TO COMPUTE RELEASE POINT

THE TIME NOW IS: 68.9 SECONDS_
```

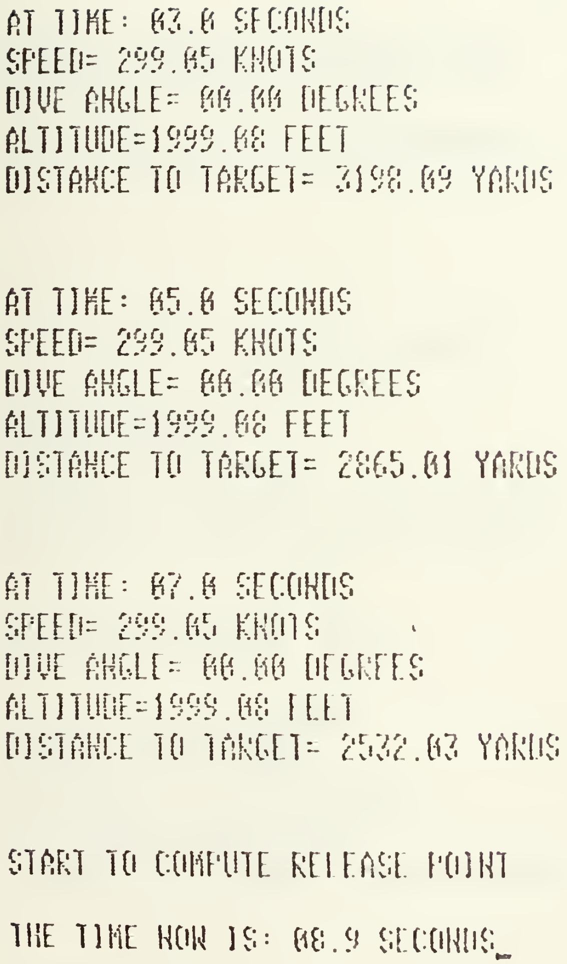
```
START TO COMPUTE RELEASE POINT

THE TIME NOW IS: 08.9 SECONDS



THE TIME NOW IS: 10.4 SECONDS

RELEASE POINT:
TIME=10.7 SECONDS
DISTANCE TO TARGET=1864.06 YARDS
DOWN RANGE TRAVEL=1870.00 YARDS

FROM THE TABLE OF DATA:

AT TIME: 10.7 SECONDS
SPEED= 470.00 KNOTS
DIVE ANGLE= -26.02 DEGREES
ALTITUDE=3847.61 FEET
DISTANCE TO TARGET= 1863.06 YARDS


FOR THIS POSITION:
```



```
RELEASE POINT:
TIME=10.7 SECONDS
DISTANCE TO TARGET=1864.06 YARDS
DOWN RANGE TRAVEL=1870.08 YARDS

FROM THE TABLE OF DATA:

AT TIME: 10.7 SECONDS
SPEED= 470.00 KNOTS
DIVE ANGLE= -26.02 DEGREES
ALTITUDE=3847.01 FEET
DISTANCE TO TARGET= 1863.06 YARDS


FOR THIS POSITION:
DOWN RANGE TRAVEL=1887.02 YARDS


BEGINING OF A RUN:
THE PROGRAM HAS A TABLE OF AIRCRAFT POSITIONS FOR 4 DIFFERENT RUNS,
USING THE BOMB MK82 UNRETARDED,IDNO=10.
ENTER A RUN NUMBER FROM 1 TO 4.

RUN=_
```

RUN NUMBER THREE:

```
BEGINING OF A RUN:
THE PROGRAM HAS A TABLE OF AIRCRAFT POSITIONS FOR 4 DIFFERENT RUNS,
USING THE BOMB MK82 UNRETARDED,IDNO=10.
ENTER A RUN NUMBER FROM 1 TO 4.

RUN=3

INPUT WEAPON ID NUMBER AFTER   '='
IDNO HAS TO BE FROM 1 TO 28.
IF YOU WANT TO QUIT ,TYPE CONTROL-C.

IDNO= 10

EXECUTION WILL BEGIN WHEN CLOCK=0

AT TIME: 01.0 SECONDS
SPEED= 299.05 KNOTS
DIVE ANGLE= 00.00 DEGREES
ALTITUDE=1999.08 FEET
DISTANCE TO TARGET= 3532.07 YARDS


AT TI
```



```
AT TIME: 03.0 SECONDS
SPEED= 299.05 KNOTS
DIVE ANGLE= 00.00 DEGREES
ALTITUDE=1999.08 FEET
DISTANCE TO TARGET= 3198.09 YARDS


AT TIME: 05.0 SECONDS
SPEED= 299.05 KNOTS
DIVE ANGLE= 00.00 DEGREES
ALTITUDE=1999.08 FEET
DISTANCE TO TARGET= 2865.01 YARDS


AT TIME: 07.0 SECONDS
SPEED= 299.05 KNOTS
DIVE ANGLE= 00.00 DEGREES
ALTITUDE=1999.08 FEET
DISTANCE TO TARGET= 2532.03 YARDS


START TO COMPUTE RELEASE POINT

THE TIME NOW IS: 08.9 SECONDS_
```

```
START TO COMPUTE RELEASE POINT

THE TIME NOW IS: 08.9 SECONDS




THE TIME NOW IS: 10.1 SECONDS

RELEASE POINT:
TIME=11.0 SECONDS
DISTANCE TO TARGET=1863.06 YARDS
DOWN RANGE TRAVEL=1860.05 YARDS

FROM THE TABLE OF DATA:

AT TIME: 11.0 SECONDS
SPEED= 299.05 KNOTS
DIVE ANGLE= 00.00 DEGREES
ALTITUDE=1999.08 FEET
DISTANCE TO TARGET= 1865.07 YARDS


FOR THIS POSITION:
_
```

```
RELEASE POINT:
TIME=11.0 SECONDS
DISTANCE TO TARGET=1863.06 YARDS
DOWN RANGE TRAVEL=1860.05 YARDS

FROM THE TABLE OF DATA:

AT TIME: 11.0 SECONDS
SPEED= 299.65 KNOTS
DIVE ANGLE= 00.00 DEGREES
ALTITUDE=1999.88 FEET
DISTANCE TO TARGET= 1865.07 YARDS


FOR THIS POSITION:
DOWN RANGE TRAVEL=1860.05 YARDS


BEGINING OF A RUN:
THE PROGRAM HAS A TABLE OF AIRCRAFT POSITIONS FOR 4 DIFFERENT RUNS,
USING THE BOMB MK82 UNRETARDED,IDNO=10.
ENTER A RUN NUMBER FROM 1 TO 4.

RUN=
```

```
C    ****************************************************
C    *                                                  *
C    *                 FORTRAN SIMULATION               *
C    *                                                  *
C    ****************************************************
C
C    ****************************************************
C    *                                                  *
C    *   MODULES' FUNCTIONS:                            *
C    *                                                  *
C    *   MAIN  : GOVERNS THE LOGIC OF THE WHOLE         *
C    *           SIMULATION.                            *
C    *   INIT  : PERFORMS THE INITIALIZATION FOR EACH   *
C    *           PARTICULAR RUN.                        *
C    *   ICHECK: CHECKS IF THERE IS SOLUTION TO THE     *
C    *           PROBLEM. IF SO,THEN CHECKS TO SEE      *
C    *           IF THERE IS TIME TO COMPUTE AND        *
C    *           FINALLY,IF THE A/C WILL BE INSIDE      *
C    *           THE DANGER ZONE AT TIME OF IMPACT.     *
C    *   DSPLAY: OUTPUTS THE RESULTS.                   *
C    *   POS   : CALCULATES THE AIRCRAFT POSITION AND   *
C    *           UPDATES ITS AIRSPEED.                  *
C    *   BAL   : INTERFACES THE MAIN MODULE WITH THE    *
C    *           BALLISTICS MODULES.                    *
C    *   SETDAT: INITIALIZATION PROCESS FOR BALLISTICS  *
C    *           COMPUTATIONS.                          *
C    *   DECODE: SET PARAMETERS ACCORDING TO WEAPON     *
C    *           SELECTED.                              *
C    *   TRAJ  : CALCULATES THE DOWNRANGE TRAVEL        *
C    *           OF SELECTED WEAPON.                    *
C    *   RUNGE : AUXILIARY MODULE OF TRAJ.              *
C    *   DERIV : AUXILIARY MODULE OF TRAJ.              *
C    *   INTER : COMMANDS THE INTERSECTION SEARCH       *
C    *           PATTERN FOR A/C AND DOWNRANGE TRAVEL   *
C    *           CURVES.                                *
C    *   COMP  : DOES A STEPWISE COMPARISON BETWEEN     *
C    *           THE A/C AND DOWNRANGE TRAVEL CURVES    *
C    *           LOOKING FOR THE INTERSECTION.          *
C    *   ORTFIT: EXTRAPOLATES VALUES FOR THE A/C AND    *
C    *           DOWNRANGE TRAVEL CURVES.               *
C    *                                                  *
C    ****************************************************
C
C    ****************************************************
C    *                                                  *
C    *   VARIABLES:                                     *
C    *   VKTS - AIR SPEED OF A/C.                       *
C    *   HV   - HORIZONTAL SPEED OF A/C.                *
C    *   HVA  - PREVIOUS VALUE OF HV.                   *
C    *   VV   - VERTICAL   SPEED OF A/C.                *
C    *   VVA  - PREVIOUS VALUE OF VV.                   *
C    *   XAT  - HORIZONTAL DISTANCE FROM A/C TO TARGET. *
C    *   ALT  - ALTITUDE OF A/C.                        *
C    *   TIME - REAL TIME CLOCK.                        *
C    *   FTIME- SAVED VALUE OF TIME FOR THE DSPLAY      *
C    *          MODULE.                                 *
C    *   HTIME- PREVIOUS VALUE OF TIME.                 *
C    *   DTNAV- TIME SPENT BY NAV COMPUTER.             *
C    *   DTBAL- TIME SPENT BY BAL COMPUTER.             *
C    *   DEG  - DIVE ANGLE OF A/C.                      *
C    *   YT   - TARGET ALTITUDE.                        *
C    *   STIME- STARTING TIME OF SIMULATION.            *
C    *   ABV  - ABSOLUTE VALUE OF DEG.                  *
C    *   SABV - SIN OF ABV.                             *
C    *   IDNO - IDENTIFICATION OF WEAPON.               *
C    *   XAC  - ARRAY OF A/C POSITIONS.                 *
C    *   RTIME- ARRAY OF TIMES CORRESPONDING TO THESE   *
C    *          POSITIONS.                              *
C    *   XDROP- ARRAY OF DOWNRANGE TRAVEL VALUES.       *
C    *   DTNAV- TIME SPENT BY NAV COMPUTER FOR THE      *
C    *          A/C POSITION CALCULATION.               *
C    *   DTBAL- TIME SPENT BY BAL COMPUTER TO CALCULATE *
```

```
C    *          THE DOWNRANGE TRAVEL FOR THIS A/C       *
C    *          POSITION.                               *
C    *   DTE1 - TIME SPENT BY THE EXEC COMPUTER TO      *
C    *          TRANSFER DATA FROM THE NAVIGATIONAL     *
C    *          COMPUTER.                               *
C    *   DTE2 - TIME SPENT BY THE EXEC COMPUTER TO      *
C    *          TRANSFER DATA FROM THE BALLISTICS       *
C    *          COMPUTER.                               *
C    *   DTE3 - TIME SPENT BY THE EXEC COMPUTER TO      *
C    *          COMPUTE THE RELEASE POINT.              *
C    *   DTE4 - TIME SPENT BY THE EXEC COMPUTER TO      *
C    *          EXECUTE CHECK NO. 2.                    *
C    *   DTE5 - TIME SPENT BY THE EXEC COMPUTER TO      *
C    *          TRANSFER DATA FROM THE NAVIGATIONAL     *
C    *          COMPUTER AND EXECUTE CHECK NO. 4.       *
C    *   DTE6 - TIME SPENT BY THE EXEC COMPUTER TO      *
C    *          SAVE AND RESTORE VALUES WHEN CALLING    *
C    *          THE  NAVIGATIONAL COMPUTER AND EXECUTE  *
C    *          CHECK NO. 3.                            *
C    *   TC   - 6*DTBAL+DTE1+DTE2+DTE3+DTE4+DTE6.       *
C    *   TARP(2)- TIME AT RELEASE POINT.                *
C    *   XARP(2)- DISTANCE TO TARGET AT RELEASE POINT.  *
C    *   TMV  - TIME IT TAKES TO THE A/C TO START       *
C    *          EVASION.                                *
C    *   ATOL - LIMIT ANGLE BETWEEN A/C PATH AND        *
C    *          THE SIGHT LINE OF TARGET.               *
C    *   RLIM - RADIUS OF A HEMISPHERE ,INSIDE          *
C    *          WHICH THE A/C WOULD BE IN DANGER        *
C    *          AT THE MOMENT OF IMPACT.                *
C    *   ALIM - LIMITING ALTITUDE FOR THE A/C.          *
C    *                                                  *
C    *   UNITS:                                         *
C    *   DISTANCES - NAUTICAL MILES.                    *
C    *   ALTITUDES - FEET.                              *
C    *   ANGLES    - DEGREES.                           *
C    *   TIME      - SECONDS.                           *
C    *   SPEED     - KNOTS.                             *
C    *                                                  *
C    *   THE VALUES OF DTE1,DTE2,DTE3,DTE4,TMV,         *
C    *   ATOL,RLIM AND ALIM ARE ASSUMED IN THIS         *
C    *   SIMULATION. MOST OF THESE ARE  DEPENDANT       *
C    *   UPON THE TYPE OF WEAPON .                      *
C    *                                                  *
C    ****************************************************
C
C    ****************************************************
C    *                                                  *
C    *   MAIN PROGRAM  -    IT SIMULATES THE            *
C    *   OPERATION OF THE WHOLE SYSTEM.                 *
C    *                                                  *
C    ****************************************************
C
      INTEGER*4 ITB(12)/12*0/
      REAL*4 RTB(28)/28*0.0/
      EQUIVALENCE (TITLE,RTB(5))
      REAL*8 TITLE(12)/'CURVES O','F XAC AN','D XDROP,',
     1'VS. TIME'/
      COMMON/DSP/FTIME,XP,TP,AARP
      COMMON/CHK/ATOL,TC,TMV,RLIM,ALIM
      COMMON/SAV/SHVA,SXAT
      COMMON/TAD/T,X
      COMMON/CLK/TIME,STIME
      COMMON/INI/SABV, HVA, VVA, HTIME, XAT,ABV
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
      COMMON/VEC/RTIME(4),XDROP(4),XAC(4)
      COMMON/DTS/DTNAV,DTBAL,DTE1,DTE2,DTE3,DTE4,DTE5,DTE6
      DIMENSION XARP(2),TARP(2)
C
C    ****************************************************
C    *                                                  *
C    *   INITIALIZATION.                                *
C    *                                                  *
```

```
C     ***********************************************************
C
   16  CALL INIT
       IF (DEG.GT.60.0) GO TO 17
       WRITE(6,2)
    2  FORMAT('1')
       WRITE(6,3)
    3  FORMAT(///)
    4  FORMAT(/)
       WRITE(6,8)
       WRITE(6,5)
       WRITE(6,31)
       WRITE(6,12)VKTS,DTNAV
       WRITE(6,14)XAT,DTBAL
       WRITE(6,13)ALT,DTE1
       WRITE(6,15)DEG,DTE2
       WRITE(6,11)TIME,DTE3
       WRITE(6,30)YT,IDNO
       WRITE(6,33)TC,TMV
       WRITE(6,32) ATOL,RLIM
       WRITE(6,34)ALIM,DTE4
       WRITE(6,35)DTE5,DTE6
       WRITE(6,5)
       WRITE(6,8)
   31  FORMAT(25X,'*',14X,'INPUT DATA:',14X,'*')
   12  FORMAT(25X,'*',3X,' VKTS=',F8.2,4X,'DTNAV=',F8.3,4X,'*
      1')
   14  FORMAT(25X,'*',3X,'  XAT=',F8.2,4X,'DTBAL=',F8.3,4X,'*
      1')
   13  FORMAT(25X,'*',3X,'  ALT=',F8.1,4X,' DTE1=',F8.3,4X,'*
      1')
   15  FORMAT(25X,'*',3X,'  DEG=',F8.1,4X,' DTE2=',F8.3,4X,'*
      1')
   11  FORMAT(25X,'*',3X,' TIME=',F8.1,4X,' DTE3=',F8.3,4X,'*
      1')
   34  FORMAT(25X,'*',3X,' ALIM=',F8.1,4X,' DTE4=',F8.3,4X,'*
      1')
   30  FORMAT(25X,'*',3X,'   YT=',F8.1,4X,' IDNO=',I8,4X,'*')
   32  FORMAT(25X,'*',3X,' ATOL=',F8.1,4X,' RLIM=',F8.1,4X,'*
      1')
   33  FORMAT(25X,'*',3X,'   TC=',F8.3,4X,'  TMV=',F8.3,4X,'*
      1')
   35  FORMAT(25X,'*',3X,' DTE5=',F8.3,4X,' DTE6=',F8.3,4X,'*
      1')
       M=1
       ICHEKK= ICHECK(M)
       IF(DEG .GE. 0.0) TIME=TIME+DTBAL
C     ***********************************************************
C     *                                                         *
C     *   ICHECK SUBROUTINE CALLS BAL- UPDATE CLOCK.            *
C     *                                                         *
C     ***********************************************************
C
       IF(ICHEKK .EQ. 1) GO TO 18
C     ***********************************************************
C     *                                                         *
C     *   IF THIS STATEMENT IS TRUE,THERE IS NO                 *
C     *   SOLUTION TO THE CURRENT DATA.                         *
C     ***********************************************************
C
       M=2
       ICHEKK= ICHECK(M)
C     ***********************************************************
C     *                                                         *
C     *   UPDATE CLOCK.                                         *
C     *                                                         *
C     ***********************************************************
       TIME=TIME+DTBAL+DTE4
       IF(ICHEKK .EQ. 2) GO TO 18
C
C     ***********************************************************
C     *                                                         *
```

```
C    *   GET THE INITIAL FOUR POSITIONS OF A/C AND                    *
C    *   CORRESPONDING DOWNRANGE TRAVELS.                             *
C    *                                                                *
C    ******************************************************************
C
      WRITE(6,8)
      WRITE(6,5)
      WRITE(6,23)
  23  FORMAT(25X,'*',6X,'PRELIMINARY  COMPUTATIONS:',7X,'*')
      WRITE(6,5)
      DO 1 I=1,4
      FTIME=TIME
C
C    ******************************************************************
C    *                                                                *
C    *   CALL POS TO GET SIMULATED A/C POSITION.                      *
C    *                                                                *
C    ******************************************************************
C
      CALL POS
      CALL DSPLAY(1)
      WRITE(6,5)
      RTIME(I)=TIME
      XAC(I)=XAT
C
C    ******************************************************************
C    *                                                                *
C    *   UPDATE CLOCK.                                                *
C    *                                                                *
C    ******************************************************************
C
      TIME=TIME+DTE1
      CALL DSPLAY(2)
      WRITE(6,5)
C
C    ******************************************************************
C    *                                                                *
C    *   CALL BAL TO GET DOWNRANGE TRAVEL FOR WEAPON                  *
C    *   IF RELEASED AT TIME CORRESPONDING TO LATEST                  *
C    *   A/C POSITION.                                                *
C    *                                                                *
C    ******************************************************************
C
      CALL BAL
      XDROP(I)=X/6080.
      CALL DSPLAY(3)
      WRITE(6,5)
C
C    ******************************************************************
C    *                                                                *
C    *   UPDATE CLOCK.                                                *
C    *                                                                *
C    ******************************************************************
C
      TIME=TIME+DTBAL+DTE2
      CALL DSPLAY(2)
  1   WRITE(6,5)
      WRITE(6,5)
      WRITE(6,8)
C
C    ******************************************************************
C    *                                                                *
C    *   CALL INTER TO OBTAIN THE RELEASE POINT                       *
C    *   EXTAPOLATED FROM THE LAST FOUR A/C POSITIONS                 *
C    *   AND CORRESPONDING DOWNRANGE TRAVELS.                         *
C    *                                                                *
C    *   TARP(1) AND XARP(1) ARE NEEDED BY THE                        *
C    *   SUBROUTINE THAT PLOTS THE GRAPHS.                            *
C    *                                                                *
C    ******************************************************************
C
      CALL INTER(XP,TP)
```

```
      TARP(1)=0.0
      XARP(1)=0.0
      TARP(2)=TP
      XARP(2)=XP
C
C     ***********************************************************
C     *                                                         *
C     *   UPDATE CLOCK.                                         *
C     *                                                         *
C     ***********************************************************
      TIME=TIME+DTE3
      ITB(1)=1
      ITB(2)=5
      ITB(3)=6
      ITB(4)=9
      RTB(1)=6.
      RTB(2)=0.9
      ITB(1)=2
      ITB(2)=2
      ITB(3)=6
      ITB(4)=9
      ITB(1)=3
      ITB(2)=1
      ITB(3)=6
      ITB(4)=9
      SAVET=TIME
      SAVEH=HTIME
      SAVEK=VKTS
      SAVEA=HVA
      SAVEV=VVA
      SAVEX=XAT
      SAVEL=ALT
      TIME=TP
C
C     ***********************************************************
C     *                                                         *
C     *   CALL PCS TO GET A/C ALTITUDE TC BE SHCWN              *
C     *   BY DSPLAY SUBROUTINE.                                 *
C     *                                                         *
C     ***********************************************************
C
      CALL POS
      AARP=ALT
      TIME=SAVET
C     ***********************************************************
C     *                                                         *
C     *   UPDATE CLOCK.                                         *
C     *                                                         *
C     ***********************************************************
      TIME=TIME+DTE6
      HTIME=SAVEH
      VKTS=SAVEK
      HVA=SAVEA
      VVA=SAVEV
      XAT=SAVEX
      ALT=SAVEL
      WRITE(6,8)
      CALL DSPLAY(4)
      CALL DSPLAY(2)
      WRITE(6,5)
      WRITE(6,8)
      M=3
      ICHEKK= ICHECK(M)
      IF(ICHEKK .EQ. 3) GO TO 18
C
C     ***********************************************************
C     *                                                         *
C     *   FRCM NOW ON,THE A/C POSITIONS AND                     *
C     *   CORRESPONDING DOWNRANGE TRAVEL OF WEAPON              *
C     *   ARE UPDATED,TOGETHER WITH A NEW RELEASE               *
C     *   POINT CALCULATION.                                    *
C     *   THIS PROCESS GOES ON UNTIL THERE IS NO                *
```

```
C     *   MORE TIME TO COMPUTE THE SOLUTION.                       *
C     *                                                            *
C     **************************************************************
C
       J=3
  43   IF(MOD(J,3) .NE. 0) GO TO 6
       WRITE(6,2)
       WRITE(6,3)
       GO TO 7
  6    WRITE(6,4)
  7    DO 42 I=1,3
       XAC(I)=XAC(I+1)
       XDROP(I)=XDROP(I+1)
  42   RTIME(I)=RTIME(I+1)
       RTIME(4)=TIME
       M=4
       ICHEKK=ICHECK(M)
       IF(ICHEKK .EQ. 4) GO TO 18
       CALL POS
       XAC(4)=XAT
       WRITE(6,8)
  8    FORMAT(25X,'*****************************************'
      1)
       WRITE(6,5)
  5    FORMAT(25X,'*',39X,'*')
       FTIME=TIME
       CALL DSPLAY(1)
C
C     **************************************************************
C     *                                                            *
C     *   UPDATE CLOCK.                                            *
C     *                                                            *
C     **************************************************************
C
       TIME=TIME+DTE5
       WRITE(6,5)
       CALL DSPLAY(2)
       WRITE(6,5)
       CALL BAL
       XDROP(4)=X/6080.
C
C     **************************************************************
C     *                                                            *
C     *   NOW THE ARRAYS ARE UPDATED.                              *
C     *                                                            *
C     **************************************************************
C
       CALL DSPLAY(3)
       WRITE(6,5)
C
C     **************************************************************
C     *                                                            *
C     *   UPDATE CLOCK.                                            *
C     *                                                            *
C     **************************************************************
C
       TIME=TIME+DTBAL+DTE2
       CALL DSPLAY(2)
       CALL INTER(XP,TP)
       TARP(2)=TP
       XARP(2)=XP
C     **************************************************************
C     *                                                            *
C     *   UPDATE CLOCK.                                            *
C     *                                                            *
C     **************************************************************
       TIME=TIME+DTE3
       SAVET=TIME
       SAVEH=HTIME
       SAVEK=VKTS
       SAVEA=HVA
       SAVEV=VVA
```

```
      SAVEX=XAT
      SAVEL=ALT
      TIME=TP
C
C     ************************************************************
C     *                                                          *
C     *   CALL PCS TO GET A/C ALTITUDE TO BE SHOWN               *
C     *   BY DSPLAY SUBROUTINE.                                  *
C     *                                                          *
C     ************************************************************
C
      CALL POS
      AARP=ALT
      TIME=SAVET
C     ************************************************************
C     *                                                          *
C     *   UPDATE CLOCK.                                          *
C     *                                                          *
C     ************************************************************
      TIME=TIME+DTNAV
      HTIME=SAVEH
      VKTS=SAVEK
      HVA=SAVEA
      VVA=SAVEV
      XAT=SAVEX
      ALT=SAVEL
      CALL DSPLAY(4)
      CALL DSPLAY(2)
      WRITE(6,5)
      WRITE(6,8)
      M=3
      ICHEKK=ICHECK(M)
      IF(ICHEKK .EQ. 3) GO TO 18
      J=J+1
      GO TO 43
   18 WRITE(6,4)
      GO TO(20,21,22,21),ICHEKK
   20 CALL DSPLAY(5)
      GO TO 40
   21 CALL DSPLAY(6)
      GO TO 40
   22 CALL DSPLAY(7)
   40 GO TO 16
   17 STOP
      END
      SUBROUTINE INIT
C     ************************************************************
C     *                                                          *
C     *   INITIALIZATION PROCESS.                                *
C     *                                                          *
C     ************************************************************
      COMMON/CHK/ATOL,TC,TMV,RLIM,ALIM
      COMMON/CLK/TIME,STIME
      COMMON/INI/SABV, HVA, VVA, HTIME, XAT,ABV
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
      COMMON/DTS/DTNAV,DTBAL,DTE1,DTE2,DTE3,DTE4,DTE5,DTE6
      READ(5,1)DEG,VKTS,XAT,ALT,YT,IDNO,TIME
    1 FORMAT(7G10.0)
      READ(5,2)DTNAV,DTBAL,DTE1,DTE2,DTE3,DTE4,DTE5,DTE6
    2 FORMAT(8F6.3)
      READ(5,3)ATOL,RLIM,ALIM,TMV
    3 FORMAT(4G5.0)
      STIME=TIME
      AEV=ABS(DEG*0.017453)
      HVA=VKTS*COS(ABV)
      SABV=SIN(ABV)
      IF (DEG.GT.0.0) SABV=-SABV
      VVA=VKTS*SABV
      HTIME=TIME
      TC=6.*DTBAL+DTE1+DTE2+DTE3+DTE4+DTE6
      RETURN
      END
```

```
      FUNCTION ICHECK(L)
C     ****************************************************************
C     *                                                              *
C     *    CHECKS:IF THERE IS A SOLUTION TO THE PROBLEM              *
C     *           IF THERE IS TIME TO COMPUTE IT.                    *
C     *           IF THE A/C WILL BE IN DANGER ZONE                  *
C     *                                                              *
C     ****************************************************************
      COMMON/DSP/FTIME,XP,TP,AARP
      COMMON/CHK/ATOL,TC,TMV,RLIM,ALIM
      COMMON/SAV/SHVA,SXAT
      COMMON/TAD/T,X
      COMMON/CLK/TIME,STIME
      COMMON/INI/SABV, HVA, VVA, HTIME, XAT,ABV
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
      COMMON/VEC/RTIME(4),XDROP(4),XAC(4)
      COMMON/DTS/DTNAV,DTBAL,DTE1,DTE2,DTE3,DTE4,DTE5,DTE6
      ICHECK=0
      GO TO (1,2,3,4),L
    1 IF(DEG .GE. 0.0) GO TO 5
      ALTM=ALT/6080.
      REL=ALTM/XAT
      ALF=ATAN(REL)
      ATOLR=ATOL*0.017453
      IF(ABV .GE. (ALF-ATOLR)) GO TO 6
      AL=ALIM/6080.
      TLIM=(ALTM-AL)*3600./VVA
      IF(TLIM .LT. (TC-DTBAL)) GO TO 6
      RETURN
    5 CALL BAL
      IF(XAT .LT. (X/6080.)) GO TO 6
      RETURN
    6 ICHECK=1
      RETURN
    2 TOC=TC
      IF(DEG .LT. 0.0) TOC=TC-DTBAL
      DTOL=HVA*TOC/3600.
C     ****************************************************************
C     *                                                              *
C     *    HERE THE DOWNRANGE TRAVEL IS CALCULATED FOR               *
C     *    THE       TIME OF COMPUTATION OF THE FOUR                 *
C     *    POINTS. THIS IS DONE,TO HAVE THE INFORMATION              *
C     *    AS SOON AS POSSIBLE,IF THERE IS GOING TO BE               *
C     *    TIME TO  FIND   THE SOLUTION.                             *
C     *                                                              *
C     *    DTOL- DISTANCE THE A/C TRAVELS DURING THE                 *
C     *          COMPUTATION OF THE FOUR POINTS AND                  *
C     *          EXTRAPOLATION OF THE SOLUTION.                      *
C     *    XDR - ESTIMATED DOWNRANGE TRAVEL AT RELEASE               *
C     *          POINT.                                              *
C     *    XAT - DISTANCE FROM TARGET AT PRESENT TIME.               *
C     *                                                              *
C     ****************************************************************
      SAVET=TIME
      SAVEH=HTIME
      SAVEK=VKTS
      SAVEA=HVA
      SAVEV=VVA
      SAVEX=XAT
      SAVEL=ALT
      TIME=TOC
      CALL POS
      CALL BAL
      XDR=X/6080.
      TIME=SAVET
      HTIME=SAVEH
      VKTS=SAVEK
      HVA=SAVEA
      VVA=SAVEV
      XAT=SAVEX
      ALT=SAVEL
      IF((XDR+DTOL) .GT. XAT) ICHECK=2
```

```
      RETURN
 3    SQ1=XP*XP
      AARPM=AARP/6080.
      SQ2=AARPM*AARPM
      SQ3=(RLIM+VKTS*TMV/3600.)*(RLIM+VKTS*TMV/3600.)
      IF(SQ3 .GT. (SQ1+SQ2)) ICHECK=3
      RETURN
 4    DTOL=HVA*2.*(DTBAL+DTE2+DTE3+DTE5+DTE6)/3600.
C     **************************************************
C     *                                                *
C     *   HERE ,INSTEAD OF USING BAL TO COMPUTE THE    *
C     *   DOWNRANGE TRAVEL,IT IS LINEARLY EXTRAPOLATED *
C     *   FROM LAST TWO VALUES. THIS IS DONE TO SAVE   *
C     *   TIME                                         *
C     *                                                *
C     **************************************************
      DSR=(XDROP(3)-XDROP(2))/(RTIME(3)-RTIME(2))
      RTM=RTIME(4)-RTIME(3)
      XDR=XDROP(3)+DSR*RTM
      IF(( XDR   +DTOL) .GT.  XAT) ICHECK=4
      RETURN
      END
      SUBROUTINE DSPLAY(I)
C     **************************************************
C     *                                                *
C     *   OUTPUTS THE RESULTS.                         *
C     *                                                *
C     **************************************************
      COMMON/TAD/T,X
      COMMON/DSP/FTIME,XP,TP,AARP
      COMMON/CHK/ATOL,TC,TMV,RLIM,ALIM
      COMMON/CLK/TIME,STIME
      COMMON/INI/SABV, HVA, VVA, HTIME, XAT,ABV
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
      COMMON/VEC/RTIME(4),XDROP(4),XAC(4)
      GO TO(1,2,3,4,5,6,7),I
 1    WRITE(6,24)FTIME
 24   FORMAT(25X,'*',6X,'AT TIME=',F6.3,' :',17X,'*')
      WRITE(6,25)XAT,ALT
 25   FORMAT(25X,'*',6X,'XAT=',F6.2,2X,'ALT=',F8.1,9X,'*')
      GO TO 10
 2    WRITE(6,27)TIME
 27   FORMAT(25X,'*',6X,'THE TIME NOW IS:',F8.3,9X,'*')
      GO TO 10
 3    WRITE(6,24)FTIME
      XIS=X/6080.
      WRITE(6,26) XIS
 26   FORMAT(25X,'*',6X,'XDROP=',F6.2,21X,'*')
      GO TO 10
 4    WRITE(6,11)
 8    FORMAT(25X,'*****************************************'
 11   FORMAT(25X,'*',39X,'*')
      WRITE(6,9)
 9    FORMAT(25X,'*',2X,'RELEASE POINT:',23X,'*')
      WRITE(6,14)XP
 14   FORMAT(25X,'*',2X,'HORIZ. DIST. TO TARGET:',3X,F5.3,
     1'  NM', 2X,'*')
      WRITE(6,12)TP
 12   FORMAT(25X,'*',2X,'TIME',18X,':',2X,F6.3,' SEC',2X,
     1'*')
      WRITE(6,13)AARP
 13   FORMAT(25X,'*',2X,'A/C ALTITUDE',10X,':',1X,F7.1,'  FT
     1,2X,'*')
      WRITE(6,11)
      GO TO 10
 5    WRITE(6,8)
      WRITE(6,11)
      WRITE(6,19)
 19   FORMAT(25X,'*',6X,'NO SOLUTION FOR THESE DATA:',
     16X,'*')
      WRITE(6,11)
      WRITE(6,8)
```

```
      WRITE(6,69)
  69  FORMAT('1')
      GO TO 10
  6   WRITE(6,8)
      WRITE(6,11)
      WRITE(6,20)
  20  FORMAT(25X,'*',6X,'NOT ENOUGH TIME TO COMPUTE.',6X,'*'
     1)
      WRITE(6,11)
      WRITE(6,8)
      GO TO 10
  7   WRITE(6,8)
      WRITE(6,11)
      WRITE(6,21)
  21  FORMAT(25X,'*',6X,'THE A/C WILL BE INSIDE THE',7X,'*')
      WRITE(6,22)
  22  FORMAT(25X,'*',6X,'DANGER ZONE AT TIME OF HIT',7X,'*')
      WRITE(6,11)
      WRITE(6,8)
  10  RETURN
      END
      SUBROUTINE POS
C    *************************************************
C    *                                               *
C    *   THIS SUBROUTINE UPDATES VALUE  OF VKTS AND  *
C    *   COMPUTES VALUES OF HV,VV,XAT AND ALT.       *
C    *   IT REPRESENTS THE NAVIGATIONAL COMPUTER.    *
C    *                                               *
C    *************************************************
      COMMON/CLK/TIME,STIME
      COMMON/INI/SABV, HVA, VVA, HTIME, XAT,ABV
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
      DT= TIME-HTIME
      VKTS=VKTS+32.174*DT*SABV
      VV=VKTS*SABV
      HV=VKTS*COS(ABV)
      HMV=(HV+HVA)/2.
      VMV=( VV+VVA)/2.
      XAT=XAT-HMV*DT/3600.
      ALT=ALT-VMV*DT*6080./3600.
      HTIME=TIME
      HVA=HV
      VVA=VV
      RETURN
      END
      SUBROUTINE BAL
C    *************************************************
C    *                                               *
C    *   THIS SUBROUTINE CALCULATES TIME OF FLIGHT   *
C    *   AND DOWN RANGE TRAVEL OF THE BOMBS.         *
C    *                                               *
C    *************************************************
      COMMON/AREA/ CT(2,2), CC(3,3,2), IREF,
     1CFORM1, DKG1, DMAX, ITYPE, IBOTH, DM1,CFORM2, DM2,
     2DKG2, DS, SL, FN, VMUZ, RAD, FRACT, VE, THETA, U, DEL,
     3V, VXA, VYA, CF, DM, DKG, VX, VY, TH, Y, YA,
     4VXO, YO, VYO, AN1, AN2, AP1, AP2, IREG, RHO, DTI, DTV
     5    ,G, A, AA,VYK,D,CKDG,MSTG
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
C CALL SETDAT TO INITIALIZE THE CONSTANTS
      CALL SETDAT
C CALL DECODE TO INITIALIZE THE BOMB COEFFICIENTS
      CALL DECODE
C CALL TRAJ TO CALCULATE THE SOLUTION TO THE DIFF. EQUATIONS
      CALL TRAJ
      RETURN
      END
      SUBROUTINE SETDAT
      COMMON/AREA/ CT(2,2), CC(3,3,2), IREF,
     1CFORM1, DKG1, DMAX, ITYPE, IBOTH, DM1,CFORM2, DM2,
     2DKG2, DS, SL, FN, VMUZ, RAD, FRACT, VE, THETA, U, DEL,
```

```
     3V, VXA, VYA, CF, DM, DKG, VX, VY, TH, Y, YA,
     4VXO, YO, VYO, AN1, AN2, AP1, AP2, IREG, RHO, DTI, DTV
     5     ,G, A, AA,VYK,D,CKDG,MSTG
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
      G= 32.174
      RAD= 0.0174533
      A= 0.7
      AA= 0.5/A
      VYK = -5.0
      FRACT= 0.5
      DS = 0.0
      CFORM1= 0.0
      CFORM2= 0.0
      DM1= 0.0
      DM2= 0.0
      DKG1= 0.0
      DKG2= 0.0
      VMUZ= 0.0
      VE= 0.0
      SL= 0.0
      DMAX= 5.0
      ITYPE= -1
      FN= 0.0
      IBOTH= 1
      T= 0.0
      U= VKTS     *1.6878
      DEL= ATAN2(VE,U)
      V= SQRT(U*U+VE*VE)
      THETA= DEG        *RAD
      VXA= (V+VMUZ)* COS(THETA-DEL)
      VYA= (V+VMUZ)* SIN(THETA-DEL)
      RETURN
      END
      SUBROUTINE DECODE
      COMMON/AREA/ CT(2,2), CC(3,3,2), IREF,
     1CFORM1, DKG1, DMAX, ITYPE, IBOTH, DM1,CFORM2, DM2,
     2DKG2, DS, SL, FN, VMUZ, RAD, FRACT, VE, THETA, U, DEL,
     3V, VXA, VYA, CF, DM, DKG, VX, VY, TH, Y, YA,
     4VXO, YO, VYO, AN1, AN2, AP1, AP2, IREG, RHO, DTI, DTV
     5     ,G, A, AA,VYK,D,CKDG,MSTG
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19
     1,20,21,22,23,24,25,26,27,28), IDNO
C WEAPON CONSTANTS FOR THE MK 43 UNRETARDED
    1 IREF= 4
      DKG1= 2.5506E-03
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 57 UNRETARDED
    2 IREF= 4
      DKG1= 6.2994E-03
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 61 UNRETARDED
    3 IREF= 4
      DKG1= 4.01E-03
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 116 WETEYE
    4 IREF= 2
      DMAX = 3.0
      CFORM1= 3.9235E-03
      DKG1= 2.754E-03
      DTI = 2.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 76 WITH LUG
    5 IREF= 2
      DMAX = 3.0
      CFORM1= 3.9077E-03
      DKG1= 6.3648E-03
```

```
      DTI = 1.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 77 FIREBOMB
    6 IREF= 4
      DMAX = 2.
      DKG1= 0.021266
      DTI = 1.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 81
    7 IREF= 1
      CFORM1= 2.5704
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 81 SNAKEYE UNRETARDED
    8 IREF= 4
      DMAX = 3.0
      DKG1= 9.767E-03
      DTI = 2.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 82 MECH FUZE
    9 IREF= 1
      CFORM1= 2.064
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 82 ELEC FUZE
   10 IREF= 1
      CFORM1= 1.4932
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 83 MECH FUZE
   11 IREF= 1
      CFORM1= 1.3431
      DTI = 1.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK83 ELEC FUZE
   12 IREF= 1
      CFORM1= 1.21
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 84
   13 IREF= 1
      CFORM1= 1.0
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 117 AL
   14 IREF= 1
      CFORM1= 3.12
      DKG1= -1.223E-03
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 86 WET SAND FILLED
   15 IREF= 1
      DMAX = 3.
      CFORM1= 3.4972
      DTI = 2.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 88 WET SAND FILLED
   16 IREF=1
      CFORM1= 1.605
      DTI = 3.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 82 SNAKEYE UNRETARDED
   17 IREF= 4
      DMAX = 3.
      DKG1= 7.329E-03
      DTI = 1.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 82 SNAKEYE RETARDED
   18 IREF= 1
      ITYPE= 1
      IBOTH= 2
      DKG1= 7.329E-03
```

```
      CFORM2= 1.6895E-02
      DM2= 0.38
      DKG2= 0.17166
      DS= 0.6617
      SL= -0.000269
      DTI = 2.0
      GO TO 31
C WEAPON CONSTANTS FOR THE SADEYE T1 = 4.0
   19 IREF= 1
      ITYPE= 1
      IBOTH= 2
      CFORM1= 2.0754
      CFORM2= 0.2217
      DS = 4.267
      DTI = 1.5
      GO TO 31
C WEAPON CONSTANTS FOR THE ROCKEYE II T1 = 4.0
   20 IREF= 1
      ITYPE= 1
      IBOTH= 2
      CFORM1= 2.2973
      DM1= 0.32
      DKG1= 8.175E-03
      CFORM2= 1.1136E-02
      DM2= 0.41
      DKG2= 0.16885
      DS = 4.06
      DTI = 2.0
      GO TO 31
C WEAPON CONSTANTS FOR THE CBU T1 = 4.0
   21 IREF= 1
      ITYPE= 1
      IBOTH= 2
      CFORM1= 2.2404
      CFORM2= 0.1178
      DS = 4.0
      DTI = 1.62
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 81 SNAKEYE RETARDED
   22 IREF= 1
      ITYPE= 1
      IBOTH= 2
      DKG1= 9.767E-03
      CFORM2= 2.30625E-02
      DM2= 0.38
      DKG2= 0.23287
      DS= 0.679
      SL= -0.000303
      DTI = 1.622
      GO TO 31
C WEAPON CONSTANTS FOR THE GUN
   23 IREF= 3
      DMAX = 1.5
      CFORM1= 2.9964
      DKG1= -0.014992
      VMUZ= 3300.0
      DTI = 0.5
      GO TO 31
C WEAPON CONSTANTS FOR THE ROCKETS
   24 IREF= 3
      ITYPE= 2
      CFORM1= 0.82
      CFORM2 = 1.0
      FN= 1746.0
      DS = 1.4225
      DTI = 1.
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 43 RETARDED 0.4 SEC DELAY
   25 IREF= 4
      ITYPE= 0
      DS= 0.98
      DKG2= 1.48
```

```
      DTI = 0.31
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 57 RETARDED 0.8 SEC DELAY
   26 IREF= 4
      ITYPE= 0
      DS= 0.89
      DKG2= 2.0
      DTI = 0.22
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 61 RETARDED 0.6 SEC DELAY
   27 IREF= 4
      ITYPE= 0
      DS= 0.89
      DKG2= 2.70
      DTI = 0.1
      GO TO 31
C WEAPON CONSTANTS FOR THE MK 106 MOD 2
   28 IREF= 2
      ITYPE= 2
      CFORM1= 0.1514
      CFORM2= 0.1514
      DS = 0.5
      DTI = 0.8
C SET THE REFERENCE DRAG CURVE COEFFICIENTS AND CUTS
   31 GO TO (32,33,34,51), IREF
   32 CC(1,1,1)= 1.572924E-03
      CC(1,2,1)= 0.0
      CC(1,3,1)= 0.0
      CC(2,1,1)= 4.678409E-02
      CC(2,2,1)=-0.109711069
      CC(2,3,1)= 6.654801E-02
      CC(3,1,1)=-0.116380157
      CC(3,2,1)= 0.217643894
      CC(3,3,1)= -9.767068E-02
      CT(1,1)= 0.834
      CT(2,1)= 0.977
      IF (IBOTH-1) 33,99,33
   33 CC(1,1,IBOTH)= 3.53503924
      CC(1,2,IBOTH)=-3.34778216
      CC(1,3,IBOTH)= 2.87262413
      CC(2,1,IBOTH)=11.2616503
      CC(2,2,IBOTH)=-27.4162512
      CC(2,3,IBOTH)= 21.7308359
      CC(3,1,IBOTH)=-23.7915472
      CC(3,2,IBOTH)= 44.2607764
      CC(3,3,IBOTH)=-14.4996046
      CT(1,IBOTH)= 0.622
      CT(2,IBOTH)= 0.385
      GO TO 51
   34 CC(1,1,1)=  0.104115
      CC(1,2,1)= -0.230347
      CC(1,3,1)=  0.167644
      CC(2,1,1)= -0.194037
      CC(2,2,1)=  0.401478
      CC(2,3,1)= -0.164612
      CC(3,1,1)=  7.33246E-02
      CC(3,2,1)= -2.03275E-02
      CC(3,3,1)=  2.44682E-03
      CT(1,1)= 1.032
      CT(2,1)= 1.30
 99   DO 100 I=1,3
      DO 100 J=1,3
      CC(I,J,2)=0.0
  100 CONTINUE
      CT(1,2)=0.0
      CT(2,2)=0.0
   51 RETURN
      END
      SUBROUTINE TRAJ
      COMMON/AREA/ CT(2,2), CC(3,3,2), IREF,
     1CFORM1, DKG1, DMAX, ITYPE, IBOTH, DM1,CFORM2, DM2,
     2DKG2, DS, SL, FN, VMUZ, RAD, FRACT, VE, THETA, U, DEL,
```

```
     3V, VXA, VYA, CF, DM, DKG, VX, VY, TH, Y, YA,
     4VXO, YO, VYO, AN1, AN2, AP1, AP2, IREG, RHO, DTI, DTV
     5    ,G, A, AA,VYK,D,CKDG,MSTG
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
C INITIALIZE THE VARIABLES FOR THE TRAJECTORY SUBROUTINE
      CF= CFORM1
      DM= DM1
      DKG= DKG1
      MSTG= 1
      X= 0.0
      T= 0.0
      VX= VXA
      VY= VYA
      TH= FN
      Y= ALT
      YA= Y
C TYPE CF DRAG
      IF (ITYPE) 2,1,1
C SET STEP SIZE FOR FIRST STAGE DRAG
    1 D= DS+SL*U
      GO TO 3
C COMPUTE STEP SIZE
    2 D = DMAX
C CALL RUNGE KUTTA SUBROUTINE
    3 CALL RUNGE
      DTV = 1/G*(VY+SQRT(VY**2+2.*G*(Y)))
      D = DTI
      IF ((IDNO.LE.17).OR.(IDNO.EQ.23)) GO TO 4
C SET THE SECOND STAGE DRAG PARAMETERS
      MSTG = 2
      IF (ITYPE.EQ.2) MSTG = 1
      DKG= DKG2
      DM= DM2
      CF = CFORM2
      TH = 0.0
    4 IF (DTV - D) 5,3,3
C SET THE STEP SIZE TO THE VACUUM DROP TIME REMAINING
    5 D= DTV
C SET THE DRAG PARAMETERS FOR THE FINAL INTEGRATION STEP
      MSTG= 2
      IF (ITYPE.EQ.2) MSTG = 1
      DKG= DKG2
      DM= DM2
      TH=0.0
      CF= CFORM2
C CALL RUNGE FOR THE FINAL INTEGRATION STEP
      CALL RUNGE
      DTV = 1/G*(VY+SQRT(VY**2+2.*G*(Y)))
C UPDATE THE TIME OF FALL AND THE DOWN RANGE TRAVEL
      T = T + DTV
      X = X +DTV*VX
      RETURN
      END
      SUBROUTINE RUNGE
      COMMON/AREA/ CT(2,2), CC(3,3,2), IREF,
     1CFORM1, DKG1, DMAX, ITYPE, IBOTH, DM1,CFORM2, DM2,
     2DKG2, DS, SL, FN, VMUZ, RAD, FRACT, VE, THETA, U, DEL,
     3V, VXA, VYA, CF, DM, DKG, VX, VY, TH, Y, YA,
     4VXO, YO, VYO, AN1, AN2, AP1, AP2, IREG, RHO, DTI, DTV
     5    ,G, A, AA,VYK,D,CKDG,MSTG
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
C INITIALIZE THE VARIABLES FOR THE RUNGE KUTTA
      AD = A*D
      YO= Y
      VXO= VX
      VYO= VY
      RHO= 2.37570E-03-Y*(6.87557E-08-Y*6.71618E-13)
      CALL DERIV
C UPDATE POSITION AND VELOCITIES
      Y= YO+AD*VY
```

```
      RHO= 2.37576E-03-Y*(6.87557E-08-Y*6.71618E-13)
      AP1= AP2
      AN1= AN2
      VX= VXO+AD*AN1
      VY= VYO+AD*AP1
      CALL DERIV
C COMPUTE TIME, POSITION AND VELOCITIES
      T= T+D
      X= X+D*(VXO+AA*(VX-VXO))
      Y= YO+D*(VYO+AA*(VY-VYO))
      VX= VXO+D*(AN1+AA*(AN2-AN1))
      VY= VYO+D*(AP1+AA*(AP2-AP1))
      RETURN
      END
      SUBROUTINE DERIV
      COMMON/AREA/ CT(2,2), CC(3,3,2), IREF,
     1CFORM1, DKG1, DMAX, ITYPE, IBOTH, DM1,CFORM2, DM2,
     2DKG2, DS, SL, FN, VMUZ, RAD, FRACT, VE, THETA, U, DEL,
     3V, VXA, VYA, CF, DM, DKG, VX, VY, TH, Y, YA,
     4VXO, YO, VYO, AN1, AN2, AP1, AP2, IREG, RHO, DTI, DTV
     5    ,G, A, AA,VYK,D,CKDG,MSTG
      COMMON/TAD/T,X
      COMMON/ACT/VKTS,ALT,DEG,YT,IDNO
C COMPUTE THE TOTAL VELOCITY AND THE MACH OF THE WEAPON
      V = SQRT(VX*VX+VY*VY)
      CM= V*(8.9544E-04+3.26E-09*Y)+DM
C DETERMINE THE REGION OF THE DRAG CURVE THAT IS APPLICABLE
      IF (CM-CT(1,MSTG)) 1,1,2
    1 IREG= 1
      GO TO 5
    2 IF (CM-CT(2,MSTG)) 3,3,4
    3 IREG= 2
      GO TO 5
    4 IREG= 3
C DO INTERMEDIATE BALLISTIC CALCULATION
    5 CKDG = DKG+CF*(CC(IREG,1,MSTG)+(CC(IREG,2,MSTG)
     1+CC(IREG,3,MSTG)*CM)*CM)
      HH= TH/V-RHO*CKDG*V
      AN2= HH*VX
      AP2= HH*VY-G
      RETURN
      END
      SUBROUTINE INTER(XARP,TARP)
C    ******************************************************
C    *                                                    *
C    *   SETS UP THE INCREMENTS OF TIME AND THE           *
C    *   DIRECTION OF SEARCH OF THE INTERSECTION.         *
C    *                                                    *
C    ******************************************************
      COMMON/VEC/RTIME(4),XDROP(4),XAC(4)
      COMMON/CLK/TIME,STIME
      COMMON/RET/RT,ZERO
      RT=RTIME(1)
      DT=1.
C            RES=1.0 MEANS THE ANSWER WAS FOUND.
      RES=COMP(DT)
      IF (RES .EQ. 1.0) GO TO 1
      DT=-0.1
      RES=COMP(DT)
      IF (RES .EQ. 1.0) GO TO 1
      DT=0.01
      RES=COMP(DT)
      IF (RES .EQ. 1.0) GO TO 1
      DT=-0.0009
      RES=COMP(DT)
  1   TARP=RT
      XARP=ZERO
      RETURN
      END
      FUNCTION COMP(DT)
C    ******************************************************
C    *                                                    *
```

```
C     *   EXECUTES THE SEARCH UNTIL THERE IS A CHANGE        *
C     *   OF SIGN OR THE INTERSECTION WAS FOUND.              *
C     *                                                       *
C     *********************************************************
      CCMMON/VEC/RTIME(4),XDROP(4),XAC(4)
      COMMON/CLK/TIME,STIME
      COMMON/RET/RT,ZERO
C     ***"ZERO",FOR THE XAC ARRAY.***
C     ***"ONE",FOR THE XDROP ARRAY.***
      CCMP=-1.0
      NIL=0
      NUM=1
      DO 1 I=1,50
      ZERO=ORTFIT(RT,NIL)
      CNE=ORTFIT(RT,NUM)
      IF (ABS(ONE-ZERO).LT.0.001) GO TO 2
      IF (DT .GT. 0.0) GO TO 3
      IF (ZERO.GT.CNE) RETURN
      GO TO 1
    3 IF (CNE.GT.ZERO) RETURN
    1 RT=RT+DT
    2 CCMP=1.0
C         COMP=1.0 MEANS THE ANSWER WAS FOUND.
      RETURN
      END
      FUNCTION ORTFIT(RT,IARRAY)
C     *********************************************************
C     *                                                       *
C     *   THIS FUNCTION   GENERATES THE NECESSARY             *
C     *   ORTHOGONAL POLYNOMIALS ,    FINDS THE               *
C     *   COEFFICIENTS OF THE POLYNOMIAL THAT                 *
C     *   REPRESENT THE DATA POINTS AND EXTRAPOLATE FOR       *
C     *   THE REQUIRED TIME.                                  *
C     *                                                       *
C     *********************************************************
      CCMMCN/VEC/RTIME(4),XDROP(4),XAC(4)
      DIMENSION ARRAY(4)
      DIMENSION C(4)
C     *** RT: REAL TIME
C         IARRAY: SWITCH BETWEEN XDROP AND XAC ***
      IF(IARPAY.EQ.0) GO TO 2
      DO 1 I=1,4
    1 ARRAY(I)=XDROP(I)
      GO TO 4
    2 DO 3 I=1,4
    3 ARRAY(I)=XAC(I)
    4 S0=4.0
      B=0.0
      DO 8 I=1,4
    8 B=B+RTIME(I)
      B0=B/S0
      S1=0.0
      DO 9 I=1,4
    9 S1=S1+(RTIME(I)-B0)**2
      B=0.0
      DO 10 I=1,4
   10 B=B+(RTIME(I))*(RTIME(I)-B0)**2
      B1=B/S1
      C1=S1/S0
      S2=0.0
      DO 11 I=1,4
   11 S2=S2+((RTIME(I)-B1)*(RTIME(I)-B0)-C1)**2
      B=0.0
      DO 111 I=1,4
  111 B=B+RTIME(I)*((RTIME(I)-B1)*(RTIME(I)-B0)-C1)**2
      B2=B/S2
      C2=S2/S1
      S3=0.0
      DC 112 I=1,4
  112 S3=S3+((RTIME(I)-B2)*((RTIME(I)-B1)*(RTIME(I)-B0)-C1)-
     1C2*(RTIME(I)-B0))**2
      D=C.0
```

```
      DO 12 I=1,4
   12 D=D+ARRAY(I)
      C(1)=D/S0
      D=0.0
      DO 13 I=1,4
   13 D=D+(ARRAY(I))*(RTIME(I)-B0)
      C(2)=D/S1
      D=0.0
      DO 14 I=1,4
   14 D=D+((ARRAY(I))*((RTIME(I)-B1)*(RTIME(I)-B0)-C1))
      C(3)=D/S2
      D=0.0
      DO 141 I=1,4
  141 D=D+(ARRAY(I))*((RTIME(I)-B2)*((RTIME(I)-B1)*(RTIME(I)
     1-B0)-C1)-C2*(RTIME(I)-B0))
      C(4)=D/S3
      ORTFIT=C(1)+C(2)*(RT-B0)+C(3)*((RT-B1)*(RT-B0)-C1)
     1 +C(4)*((RT-B2)*((RT-B1)*(RT-B0)-C1)-C2*(RT-B0))
      RETURN
      END
```

```
/***************************************************
*                                                 *
*   EXECUTIVE'S COMPUTER PROGRAM                  *
*                                                 *
***************************************************/
DECLARE NAVTAB DATA
       (000H,000H,000H,040H,000H,000H,
        040H,0EBH,0EBH,04DH,081H,0A7H,04DH,
        00AH,0C6H,0FAH,049H,0F4H,062H,
        0C5H,0E1H,03EH,04DH,0F7H,068H,04CH,
        014H,0CBH,062H,049H,0EFH,021H,
        0C5H,0D6H,08BH,04DH,0EBH,029H,04CH,
        01EH.0CFH,09EH.049H,0EAH,062H,
        0C5H,0CBH,0D2H,04DH,0DEH,097H,04CH,
        028H,0D3H,0ACH,049H,0E6H,018H,
        0C5H,0C1H,011H,04DH,0D1H,0B4H,04CH,
        032H,0D7H,08DH,049H,0E2H,03CH,
        0C5H,0B6H,04AH,04DH,0C4H,085H,04CH,
        03CH,0DBH,03EH,049H,0DEH,0C4H,
        0C5H,0ABH,07BH,04DH,0B7H,00FH,04CH,
        046H,0DEH,08EH,049H,0DBH,0A8H,
        0C5H,0A0H,0A3H,04DH,0A9H,055H,04CH,
        050H,0E2H,00EH,049H,0D8H,0E4H,
        0C5H,095H,0C4H,04DH,09BH,05CH,04CH,
        05AH,0E5H,02CH,049H,0D6H,070H,
        0C5H,08AH,0DBH,04DH,08DH,027H,04CH,
        05FH,0E7H,063H,049H,0D4H,0C8H,
        0C5H,085H,063H,04DH,085H,0F8H,04CH,
        060H,0E8H,03DH,049H,0D4H,029H,
        0C5H,084H,04BH,04DH,084H,087H,04CH,
        061H,0E8H,088H,049H,0D4H,000H,
        0C5H,083H,032H,04DH,083H,015H,04CH,
        062H,0E8H,0CDH,049H,0D3H,0C4H,
        0C5H,082H,01AH,04DH,081H,0A2H,04CH,
        063H,0E9H,015H,049H,0D3H,097H,
        0C5H,081H,002H,04DH,080H,02FH,04CH,
        064H,0E9H,059H,049H,0D3H,061H,
        0C5H,0FFH,0D2H,04CH,0FDH,078H,04BH,
        065H,0E9H,0A0H,049H,0D3H,036H,
        0C5H,0FDH,0A1H,04CH,0FAH,090H,04BH,
        066H,0E9H,0E5H,049H,0D3H.007H,
        0C5H,0FBH,06FH,04CH,0F7H,0A7H,04BH,
        067H,0EAH,02BH,049H,0D2H,0D8H,
        0C5H,0F9H,03EH,04CH,0F4H,0BDH,04BH,
        068H,0EAH,06FH,049H,0D2H,0AAH,
        0C5H,0F7H,00CH,04CH,0F1H,0D2H,04BH,
        069H,0EAH,0B2H,049H,0D2H,07EH,
        0C5H,0F4H,0D9H,04CH,0EEH,0E6H,04BH,
        06AH,0EAH,0F5H,049H,0D2H,051H,
        0C5H,0F2H,0A7H,04CH,0EBH,0F9H,04BH,
        06BH,0EBH,038H,049H,0D2H,025H,
        0C5H,0F0H,074H,04CH,0E9H,00BH,04BH,
        06CH,0EBH,07AH,049H,0D1H,0FAH,
        0C5H,0EEH,041H,04CH,0E6H,01CH,04BH,
        06DH,0EBH,0BCH,049H,0D1H,0D0H,
        0C5H,0ECH,00EH,04CH,0E3H,02CH,04BH,
        06EH,0EBH,0FDH,049H,0D1H,0A5H,
        0C5H,0E9H,0DBH,04CH,0E0H,03BH,04BH,
        06FH,0ECH,03DH,049H,0D1H,07DH,
        0C5H,0E7H,0A8H,04CH,0DDH,049H,04BH,
        070H,0ECH,07EH,049H,0D1H,054H,
        0C5H,0E5H,074H,04CH,0DAH,057H.04BH,
        071H.0ECH,0BEH,049H,0D1H,02CH,
        0C5H,0E3H,040H,04CH,0D7H,063H,04BH,
        072H,0ECH,0FDH,049H,0D1H,0C5H,
        0C5H,0E1H,00CH,04CH,0D4H,06EH,04BH,
        073H,0EDH,038H,049H,0D0H,0DFH,
        0C5H,0DEH,0D8H,04CH,0D1H,079H,04BH,
        074H,0EDH,079H,049H,0D0H,0BAH,
        0C5H,0DCH,0A4H,04CH,0CEH,083H,04BH,
        075H,0EDH,0B7H,049H,0D0H,094H,
        0C5H,0DAH,06FH,04CH,0CBH,08BH,04BH,
```

```
076H,0EDH,0F4H,049H,0D0H,06FH,
0C5H,0D8H,03AH,04CH,0C8H,093H,04BH,
077H,0EEH,031H,049H,0D0H,04BH,
0C5H,0D6H,005H,04CH,0C5H,09AH,04BH,
078H,0EEH,06DH,049H,0D0H,028H,
0C5H,CD3H,0D0H,04CH,0C2H,0A1H,04BH,
079H,0EEH,0A8H,049H,0D0H,006H,
0C5H,0D1H,09AH,04CH,0BFH,0A6H,04BH,
07AH,0EEH,0E3H,049H,0CFH,0E4H,
0C5H,0CFH,065H,04CH,0BCH,0AAH,04BH,
07BH,0EFH,01DH,049H,0CFH,0C2H,
0C5H,0CDH,02FH,04CH,0B9H,0AEH,04BH,
07CH,0EFH,058H,049H,0CFH,0A1H,
0C5H,0CAH,0F9H,04CH,0B6H,0B1H,04BH,
000H,000H,000H,040H,000H,000H,
040H,090H,025H,04EH,0F3H,080H,04CH,
00AH,0ABH,0CFH,049H,093H,0F2H,
0C6H,08AH,0B1H,04EH,0E9H,0F8H,04CH,
014H,0B1H,09FH,049H,094H,0B8H,
0C6H,085H,006H,04EH,0E0H,024H,04CH,
01EH,0B7H,042H,049H,094H,0CAH,
0C6H,0FEH,05BH,04DH,0D6H,001H,04CH,
028H,0BCH,0BBH,049H,094H,03BH,
0C6H,0F2H,05AH,04DH,0CBH,08BH,04CH,
032H,0C2H,008H,049H,093H,017H,
0C6H,0E6H,017H,04DH,0C0H,0C0H,04CH,
03CH,0C7H,02EH,049H,091H,069H,
0C6H,0D9H,0A0H,04DH,0B5H,09CH,04CH,
046H,0CCH,02EH,049H,08FH,03AH,
0C6H,CCDH,004H,04DH,0AAH,01CH,04CH,
050H,0D1H,00BH,049H,08CH,094H,
0C6H,0C0H,051H,04DH,09EH,03EH,04CH,
05AH,0D5H,0CAH,049H,089H,07CH,
0C6H,083H,093H,04DH,091H,0FCH,04CH,
05FH,CD9H,049H,049H,086H,0E7H,
0C6H,0ADH,035H,04DH,088H,0B6H,04CH,
060H,0DAH,0ADH,049H,085H,0D0H,
0C6H,0ABH,0EFH,04DH,08AH,072H,04CH,
061H,0DBH,022H,049H,085H,070H,
0C6H,0AAH,0AAH,04DH,089H,02CH,04CH,
062H,0D8H,097H,049H,085H,00DH,
0C6H,0A9H,064H,04DH,087H,0E6H,04CH,
063H,0DCH,00CH,049H,084H,0ABH,
0C6H,0A8H,01FH,04DH,086H,09EH,04CH,
064H,0DCH,082H,049H,084H,049H,
0C6H,0A6H,0DAH,04DH,085H,056H,04CH,
065H,0DCH,0F5H,049H,083H,0E2H,
0C6H,0A5H,095H,04DH,084H,00CH,04CH,
066H,0DDH,06AH,049H,083H,07DH,
0C6H,0A4H,050H,04DH,082H,0C2H,04CH,
067H,0DDH,0E1H,049H,083H,01CH,
0C6H,0A3H,00CH,04DH,081H,076H,04CH,
068H,0DEH,055H,049H.082H,0B2H,
0C6H,0A1H,0C7H,04DH,080H,029H,04CH,
069H,0DEH,0C8H,049H,082H,049H,
0C6H,0A0H,083H,04DH,0FDH,0B8H,04BH,
06AH,0DFH,03DH,049H,081H,0E3H,
0C6H,09FH,03FH,04DH,0FBH,01AH.04BH,
06BH,0DFH,0B1H,049H,081H,077H,
0C6H;09DH,0FCH,04DH,0F8H,07AH,04BH,
06CH,0E0H,024H,049H,081H,00DH,
0C6H,09CH,0B8H,04DH,0F5H,0D8H,04BH,
06DH,0E0H,09AH,049H,080H,0A0H,
0C6H,09BH,075H,04DH,0F3H,034H,04BH,
06EH,0E1H,008H,049H,080H,02FH,
0C6H,09AH,033H,04DH,0F0H,08EH,04BH,
06FH,0E1H,080H.049H,0FFH,08DH,
0C5H,098H,0F0H,04DH,0EDH,0E6H,04BH,
070H,0E1H,0F1H,049H,0FEH,0A9H,
0C5H,097H,0AEH,04DH,0EBH,038H,04BH,
071H,0E2H,068H,049H,0FDH,0D5H,
0C5H,096H,06CH,04DH,0E8H,08EH,04BH,
```

```
072H,0E2H,0D9H,049H,0FCH,0ECH,
0C5H,095H,02BH,04DH,0E5H,0E0H,04BH,
073H,0E3H,04CH,049H,0FCH,009H,
0C5H,093H,0FAH,04DH,0E3H,02FH,04BH,
074H,0E3H,0BEH,049H,0FBH,022H,
0C5H,092H,0AAH,04DH,0E0H,07BH,04BH,
075H,0E4H,033H,049H,0FAH,03FH,
0C5H,091H,069H,04DH,0DDH,0C6H,04BH,
076H,0E4H,0A4H,049H,0F9H,054H,
0C5H.090H,02AH,04DH,0DBH,00EH,04BH,
077H,0E5H,017H,049H,0F8H,068H,
0C5H,08EH,0EAH,04DH,0D8H,054H,04BH,
078H,0E5H,08AH,049H,0F7H,07EH,
0C5H,08DH,0ABH,04DH,0D5H,098H,04BH,
C79H,0E5H,0FEH,049H,0F6H,093H,
0C5H,08CH,06DH,04DH,0D2H,0DAH,04BH,
07AH,0E6H,06FH,049H,0F5H,09FH,
0C5H,08BH,02FH,04DH,0D0H,01AH,04BH,
07BH,0E6H,0E3H,049H,0F4H,0B0H,
0C5H,089H,0F2H,04DH,0CDH,057H,04BH,
07CH,0E7H,056H,049H,0F3H,0BDH,
0C5H,088H,0B5H,04DH,0CAH,092H,04BH,
000H,000H,000H,040H,000H,000H.
040H,0FAH,000H.04BH,0E7H,035H,04CH,
00AH,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0DCH,0CBH,04CH,
014H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0D2H,060H,04CH,
01EH,096H,000H,049H,000H,000H.
040H,0FAH,000H,04BH,0C7H,0F5H,04CH,
028H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0BDH,08BH,04CH,
032H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0B3H,020H,04CH,
03CH,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0A8H,0B5H,04CH,
046H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,09EH,04BH,04CH,
050H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,093H,0E0H.04CH,
05AH,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,089H,075H,04CH,
05FH,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,084H,040H,04CH,
060H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,083H,035H,04CH,
061H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,082H,02BH,04CH,
062H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,081H,020H,04CH,
063H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,080H,015H,04CH,
064H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0FEH,016H,04BH,
065H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0FCH,000H,04BH,
066H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0F9H,0EBH,04BH,
067H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0F7H,0D6H,04BH,
068H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0F5H,0C0H,04BH,
069H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0F3H,0ABH,04BH,
06AH,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0F1H,096H,04BH,
06BH,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0EFH,081H,04BH,
06CH,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0EDH,06BH,04BH,
06DH,096H,000H,049H,000H,000H.
040H,0FAH,000H,04BH,0EBH,056H,04BH,
```

```
06EH,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0E9H,041H,04BH,
06FH,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0E7H,02BH,04BH,
070H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0E5H,016H,04BH,
071H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0E3H,001H,04BH,
072H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0E0H,0EBH,04BH,
073H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0DEH,0D6H,04BH,
074H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0DCH,0C1H,04BH,
075H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0DAH,0ABH,04BH,
076H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0D8H,096H,04BH,
077H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0D6H,081H,04BH,
078H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0D4H,06BH,04BH,
079H,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0D4H,06BH,04BH,
07AH,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0D0H,041H,04BH,
07BH,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0CEH,02BH,04BH,
07CH,096H,000H,049H,000H,000H,
040H,0FAH,000H,04BH,0CCH,016H,04BH,
000H,000H,000H,040H,000H,000H,
040H,08BH,01DH,04DH,0A3H,061H,04DH,
00AH,0E1H,000H,049H,0A0H,000H,
0C4H,086H,0FDH,04DH,09BH,0B0H,04DH,
014H,0E1H,000H,049H,0A0H,000H,
0C4H,082H,0D0H,04DH,093H,0FEH,04DH,
01EH,0E1H,000H,049H,0A0H,000H,
0C4H,0FDH,07CH,04CH,08CH,04CH,04DH,
028H,0E1H,000H,049H,0A0H,000H,
0C4H,0F5H,03CH,04CH,084H,09BH,04DH,
032H,0E1H,000H,049H,0A0H,000H,
0C4H,0ECH,0FDH,04CH,0F9H,0D3H,04CH,
03CH,0E1H,000H,049H,0A0H,000H,
0C4H,0E4H,0BDH,04CH,0EAH,070H,04CH,
046H,0E1H,000H,049H,0A0H,000H,
0C4H,0DCH,07EH,04CH,0DBH,00CH,04CH,
050H,0E1H,000H,049H,0A0H,000H,
0C4H,0D4H,03EH,04CH,0CBH,0A9H,04CH,
05AH,0E1H,000H,049H,0A0H,000H,
0C4H,0CBH,0FFH,04CH,0BCH,046H,04CH,
05FH,0E1H,000H,049H,0A0H,000H,
0C4H,0C7H,0DFH,04CH,0B4H,094H,04CH,
060H,0E1H,000H,049H,0A0H,000H,
0C4H,0C7H,00CH,04CH,0B3H,00AH,04CH,
061H,0E1H,000H,049H,0A0H,000H,
0C4H,0C6H,038H,04CH,0B1H,080H,04CH,
062H,0E1H,000H,049H,0A0H,000H,
0C4H,0C5H,065H,04CH,0AFH,0F7H,04CH,
063H,0E1H,000H,049H,0A0H,000H,
0C4H,0C4H,092H,04CH,0AEH,06DH,04CH,
064H,0E1H,000H,049H,0A0H,000H,
0C4H,0C3H,0BFH,04CH,0ACH,0E3H,04CH,
065H,0E1H,000H,049H,0A0H,000H,
0C4H,0C2H,0ECH,04CH,0ABH,059H,04CH,
066H,0E1H,000H,049H,0A0H,000H,
0C4H,0C2H,019H,04CH,0A9H,0CFH,04CH,
067H,0E1H,000H,049H,0A0H,000H,
0C4H,0C1H,046H,04CH,0A8H,045H,04CH,
068H,0E1H,000H,049H,0A0H,000H,
0C4H,0C0H,072H,04CH,0A6H,0BBH,04CH,
069H,0E1H,000H,049H,0A0H,000H,
0C4H,08FH,09FH,04CH,0A5H,031H,04CH,
```

```
       06AH,0E1H,000H,049H,0A0H,000H,
       0C4H,0BEH,0CCH,04CH,0A3H,0A7H,04CH,
       06BH,0E1H,000H,049H,0A0H,000H,
       0C4H,0BDH,0F9H,04CH,0A2H,01DH,04CH,
       06CH,0E1H,000H,049H,0A0H,000H,
       0C4H,0BDH,026H,04CH,0A0H,093H,04CH,
       06DH,0E1H,000H,049H,0A0H,000H,
       0C4H,0BCH,053H,04CH,09FH,009H,04CH,
       06EH,0E1H,000H,049H,0A0H,000H,
       0C4H,0BBH,080H,04CH,09DH,080H,04CH,
       06FH,0E1H,000H,049H,0A0H,000H,
       0C4H,0BAH,0ACH,04CH,09BH,0F6H,04CH,
       070H,0E1H,000H,049H,0A0H,000H,
       0C4H,0B9H,0D9H,04CH,09AH,06CH,04CH,
       071H,0E1H,000H,049H,0A0H,000H,
       0C4H,0B9H,006H,04CH,098H,0E2H,04CH,
       072H,0E1H,000H,049H,0A0H,000H,
       0C4H,0B8H,033H,04CH,097H,058H,04CH,
       073H,0E1H,000H,049H,0A0H,000H,
       0C4H,0B7H,060H,04CH,095H,0CEH,04CH,
       074H,0E1H,000H,049H,0A0H,000H,
       0C4H,0B6H,08DH,04CH,094H,044H,04CH,
       075H,0E1H,000H,049H,0A0H,000H,
       0C4H,0B5H,0B9H,04CH,092H,0BAH,04CH,
       076H,0E1H,000H,049H,0A0H,000H,
       0C4H,0B4H,0E6H,04CH,091H,030H,04CH,
       077H,0E1H,000H,049H,0A0H,000H,
       0C4H,0B4H,013H,04CH,08FH,0A6H,04CH,
       078H,0E1H,000H,049H,0A0H,000H,
       0C4H,0B3H,040H,04CH,08EH,01CH,04CH,
       079H,0E1H,000H,049H,0A0H,000H,
       0C4H,0B2H,06DH,04CH,08CH,092H,04CH,
       07AH,0E1H,000H,049H,0A0H,000H,
       0C4H,0B1H,09AH,04CH,08BH,008H,04CH,
       07BH,0E1H,000H,049H,0A0H,000H,
       0C4H,0B0H,0C7H,04CH,089H,07FH,04CH,
       07CH,0E1H,000H,049H,0A0H,000H,
       0C4H,0AFH,0F3H,04CH,087H,0F5H,04CH);


DECLARE INIT LITERALLY '0C0H';
DECLARE DAT  LITERALLY '0D0H';
DECLARE SEND$IDNO   LITERALLY '0ABH';
DECLARE FROM$BUFFER LITERALLY '0BBH';
DECLARE EMPTY LITERALLY '0CCH';
DECLARE INTER LITERALLY '0B0H';
DECLARE  (INDX,PID) ADDRESS;
DECLARE (RUN,IDNO,RST,IBP) BYTE;
DECLARE  (IDLE,I,TI,NI,J,NUMB) BYTE;
DECLARE TABLE(80) BYTE;

DECLARE ENTER LITERALLY '3FFDH';
DECLARE RDBUFF(130) BYTE;
DECLARE FOREVER LITERALLY 'WHILE 1';
MON1:   PROCEDURE(FUNC,INFO);
        DECLARE FUNC BYTE, INFO ADDRESS;
        GO TO ENTER;
        END MON1;

MON2:   PROCEDURE(FUNC,INFO)BYTE;
        DECLARE FUNC BYTE, INFO ADDRESS;
        GO TO ENTER;
        END MON2;

PRINTCHAR: PROCEDURE(B);
        /* SEND THE ASCII CHARACTER B TO THE CONSOLE */
        DECLARE B BYTE;
        CALL MON1(2,B);
        END PRINTCHAR;

CRLF:      PROCEDURE;
        /* SEND CARRIAGE-RETURN-LINE-FEED CHARACTERS */
```

```
        CALL PRINTCHAR(0DH);
        CALL PRINTCHAR(0AH);
        END CRLF;

PRINT:  PROCEDURE(A);
        /* PRINT THE BUFFER STARTING AT ADDRESS A */
        DECLARE A ADDRESS;
        CALL MON1(9,A);
        END PRINT;

READ:   PROCEDURE;
        /* READ CONSOLE CHARACTERS INTO 'RDBUFF' */
        RDBUFF=128; /* FIRST BYTE SET TO BUFF. LENGTH*/
        CALL MON1(10,.RDBUFF);
        END READ;

TRANS:  PROCEDURE(B) BYTE;
        /* RETURNS THE ASCII CHARACTER OF THE NUMBER */
        DECLARE B BYTE;
        IF B<10 THEN RETURN B+'0';
        ELSE RETURN B-10+'A';
        END TRANS;

CONV:   PROCEDURE(B) BYTE;
        DECLARE B BYTE;
        /* RETURNS THE VALUE OF THE NUMBER CHARACTER */
        RETURN B-'0';
        END CONV;

GETC:   PROCEDURE BYTE;
        RETURN RDBUFF((IBP:=IBP+1)-1);
        END GETC;
READC:  PROCEDURE BYTE;
        DECLARE B BYTE;
        B=GETC;
        RETURN B;
        END READC;
READP:  PROCEDURE BYTE;
        DECLARE B BYTE;
        B= CONV(READC);
        /* FIRST CHARACTER IS MULTIPLIED BY 10 */
        RETURN SHL(B,3)+SHL(B,1)+CONV(READC);
        END READP;

CLOCK:    PROCEDURE BYTE;
        DECLARE T(4) BYTE, I BYTE;
        DO I=0 TO 3;
        T(I)=INPUT(6);
        END;
        DO I=0 TO 2;
        IF T(I+1)=T(I) THEN RETURN 255-T(I);
        END;
        CALL CRLF;
        CALL PRINT(.'CAN NOT READ TIME.$');
        RETURN I;
        END CLOCK;

NEGAT:  PROCEDURE(V) BYTE;/* TRUE IF NUMBER IS NEGAT.*/
        DECLARE V ADDRESS, W BASED V BYTE;
        IF (W(2) AND 80H) <> 0 THEN RETURN 1;
                              ELSE   RETURN 0;
        END NEGAT;

DECLARE DEC(4) BYTE;
MANT:   PROCEDURE(W);
        /* FILLS UP A VECTOR OF 4 INTEGERS THAT
        REPRESENTS THE CONVERSION OF THE MANTISSA
        TO DECIMAL FORM  */
        DECLARE W ADDRESS, Y BASED W BYTE;
        DECLARE (ONE,TWO,THREE,FOUR) ADDRESS,
                (AUX,TEMP,DI) ADDRESS,
                (I,J,K) BYTE;
```

```
         ONE=SHR(Y,4)*625;
         AUX=(Y AND 0FH)*3906;
         TWO=AUX/100;
         AUX=AUX-(TWO*100);
         TEMP=AUX;
         AUX=SHR(Y(1),4)*244;
      24 THREE=AUX/100;
         AUX=AUX-(THREE*100);
         TEMP=TEMP+AUX;
      25 AUX=(Y(1) AND 0FH)*15;
         FOUR=AUX/100;
         AUX=AUX-(FOUR*100);
         TEMP=TEMP+AUX+50;
         AUX=ONE+TWO+THREE+FOUR+TEMP/100;
      26 DO I=1 TO 4;
         J=4-I;
         DI=1;
         DO K=1 TO J;
         DI=10*DI;
         END;
      27 DEC(I)=AUX/DI;
         AUX=AUX-DEC(I)*DI;
      28 END:
         END MANT;

DECIM:   PROCEDURE(Q);
         /* DISPLAYS THE DECIMAL POINT AND FRACTIONAL
         PART OF VARIABLE     */
         DECLARE Q BYTE;
         CALL PRINTCHAR('.');
         CALL PRINTCHAR(TRANS(Q));
         RETURN;
         END DECIM;

INTEG:   PROCEDURE(R,N) BYTE;
         /* DISPLAYS THE INTEGER PART OF VARIABLE
         AND RETURNS THE REMAINDER. THE ARGUMENT
         IS 10 TIMES THE VARIABLE  */
         DECLARE (I,N,J,RES,V) BYTE, (R,DI) ADDRESS;
         DECLARE K BYTE;
         DO I=0 TO N-1;
         J=N-I;
         DI=10;
         DO K=1 TO J-1;
         DI=10*DI;
         END;
         V=R/DI;
         CALL PRINTCHAR(TRANS(V));
         R=R-V*DI;
         END;
         RES=R;
         RETURN RES;
         END INTEG;

PCTEN:   PROCEDURE(Y) ADDRESS;
         /* COMPUTES THE EXPONENTIAL PART OF THE
         NORMALIZED NUMBER  */
         DECLARE Y ADDRESS, X BASED Y BYTE;
         DECLARE VAL ADDRESS;
         DECLARE (N,I) BYTE;
         N=X(2) AND 3FH;
         VAL=1;
         DO I=1 TO N;
           VAL=SHL(VAL,1);  /* MULT. BY 2 */
         END;
         RETURN VAL;
         END POTEN;

INDEX:   PROCEDURE(T) BYTE;
         /* USING REAL TIME ,FINDS THE INDEX TO
         THE NAVTAB  */
         DECLARE T BYTE;
```

```
          IF T < 93 THEN RETURN T/10;
          ELSE DO;  IF T <= 95 THEN RETURN 10;
          ELSE RETURN T-95+10;
         END;
          END INDEX;

DSPLV:    PROCEDURE(P,M);
          /* THIS PROCEDURE CONVERTS THE HEXADECIMAL
             VALUES TO DECIMAL FORM IN ORDER TO BE
             DISPLAYED */
          DECLARE (P,ACUM,REM,AUX,UP) ADDRESS,
                  (DI,ONE,TWO,DN) ADDRESS,
                  (M,LOW,I,TEMP,J,N,K,L,BTE) BYTE;
          IF NEGAT(P) THEN CALL PRINTCHAR('-');
      L1  CALL MANT(P);
          ACUM=0;
          REM=0;
          AUX=POTEN(P);
          UP=AUX/10;
          LOW=AUX-(UP*10);
      L2  DO I=1 TO 4;
          AUX=UP*DEC(I);
          TEMP=LOW*DEC(I);
          J=I-1;
          DI=1;
          DO K=1 TO J;
          DI=10*DI;
          END;
          N=3-I;
          DN=1;
      L3  DO L=1 TO N;
          DN=10*DN;
          END;
          ONE=AUX/DI;
          TWO=((AUX-(ONE*DI))*10+TEMP)*DN;
          IF (I=4) THEN TWO=TWO/10;
          ACUM=ACUM+ONE;
          REM=REM+TWO;
      L4  END;
          ACUM=ACUM+REM/1000;
          REM=REM-(REM/1000)*1000;
      L5  DO I=1 TO M;
          J=M-I;
          DI=1;
          DO K=1 TO J;
          DI=10*DI;
          END;
          BTE=ACUM/DI;
      L6  CALL PRINTCHAR(TRANS(BTE));
          ACUM=ACUM-BTE*DI;
          END;
          CALL PRINTCHAR('.');
      L7  DO I=1 TO 2;
          J=4-I;
          DI=1;
          DO K=1 TO J;
          DI=10*DI;
          END;
          BTE=REM/DI;
      L8  CALL PRINTCHAR(TRANS(BTE));
          REM=REM-BTE*DI;
          END;
          END DSPLV;

DISPLAY:    PROCEDURE(IND);
          /* THIS PROCEDURE DISPLAYS THE  POSITION,
             SPEED AND DIVE ANGLE OF THE AIRCRAFT */
            DECLARE (VAX,IND) ADDRESS,  RST BYTE;
            CALL CRLF;
            CALL PRINT(.'AT TIME: $');
            VAX=NAVTAB(IND);
            RST=INTEG(VAX,2);
```

```
               CALL DECIM(RST);   CALL PRINT(.' SECONDS $');
               IND=IND+1;
               CALL CRLF;
               CALL PRINT(.'SPEED= $');
               CALL DSPLV(.NAVTAB(IND),3);
               CALL PRINT(.' KNOTS $');
               IND=IND+3;
               CALL CRLF;
               CALL PRINT(.'DIVE ANGLE= $');
               CALL DSPLV(.NAVTAB(IND),2);
               CALL PRINT(.' DEGREES $');
               IND=IND+3;
               CALL CRLF;
               CALL PRINT(.'ALTITUDE=$');
               CALL DSPLV(.NAVTAB(IND),4);
               CALL PRINT(.' FEET $');
               IND=IND+3;
               CALL CRLF;
               CALL PRINT(.'DISTANCE TO TARGET= $');
               CALL DSPLV(.NAVTAB(IND),4);
               CALL PRINT(.' YARDS $');
               CALL CRLF;
               CALL CRLF;
               RETURN;
               END DISPLAY;

     FLOAT: PROCEDURE(INN,FA);
   /* A PROCEDURE TO CONVERT AN INTEGER INN TO
   A FLOATING POINT WHOSE MANTISSA IS FOUND IN
   ADDRESSES  FA AND FA+1  AND WHOSE EXPONENT
   IS IN FA +2  */

   DECLARE (INN,FA) ADDRESS ,MANN ADDRESS,
      MANT BASED FA BYTE,
      INT BASED INN ADDRESS,
      N BYTE;

      IF INT = 0                 /* CHECK FOR ZERO INPUT */

        THEN  DO;
        MANT(2),MANT(1),MANT = 0;
        RETURN ;
        END;

      MANT(2) = 50H;             /* EXPONENT = 2**16  */
      MANN = INT;

      IF ( MANN AND 8000H) = 8000H THEN
      /* NEGATIVE INTEGER */
        DO;   MANT(2) = MANT(2) OR 80H; /* SET SIGN BIT */
              MANN    = 1-MANN ;        /* COMPLEMENT   */
        END;

      N = 1;         /* NORMALIZE MANTISSA */
      DO WHILE (( SHL(MANN,1) AND 8000H) <> 8000H);
       N=N+1;
       MANN = SHL(MANN,1);
      END;

      MANN= SHL(MANN,1);
      MANT(2) = MANT(2) - N;  /* ADJUST EXPONENT */
      MANT    = HIGH(MANN);   /* ASSIGN MANTISSA */
      MANT(1) = LOW(MANN);
      RETURN ;
   END FLOAT;
 DECLARE ZE BYTE, ZZ ADDRESS;
DECLARE YE BYTE,XE BYTE;
  /* FLOATING POINT ADD ROUTINE*/
ADD: PROCEDURE (XA,YA,PA);
     DECLARE (XA,YA,ZA) ADDRESS,
       PA ADDRESS, P BASED PA BYTE,
             (I,R2,Y2,DIGIT,SINE) BYTE,
```

```
           X BASED XA BYTE,
           Y BASED YA BYTE,
            (XX,YQ) ADDRESS;

/* PROCEDURE TO LEFT JUSTIFY MANTISSA IN BINARY */
  ADJUST: PROCEDURE;
    DO I=0 TO 15;
       IF (ZZ AND 8000H)=8000H THEN RETURN;
      ZZ=SHL(ZZ,1);
      ZE=ZE-1;
   END;   RETURN;
    END ADJUST;

   XX=SHL(DOUBLE(X),8) OR X(1);
   YQ=SHL(DOUBLE(Y),8) OR Y(1);
   R2=       X(2) AND 7FH;
   Y2=       Y(2) AND 7FH;
   DIGIT=R2-Y2;
    SINE =(X(2) AND 80H) XOR (Y(2) AND 80H);
    IF (DIGIT AND 80H)<>0      THEN DIGIT=-DIGIT;
   IF DIGIT >= 16  THEN DO;
/* VARIABLES NOT WITHIN SIGNIFICANCE RANGE */
    IF R2 > Y2 THEN DO;
     ZZ=XX;  ZE=X(2);  GO TO RET;
      END;
     ZZ=YQ;  ZE=Y(2);  GO TO RET;
      END;
    IF Y2 = R2 THEN DO;
/* EXPONENTS EQUAL IN ABSOLUTE VALUE      */
         IF YQ>XX THEN DO;
/* Y > X */
            ZE= Y(2);
              IF SINE < 80H THEN GO TO EXIT1;
                          GO TO EXIT2;
                  END;
        IF YQ<XX THEN DO;
/* X > Y */
                  ZE =X(2);
              IF SINE < 80H THEN GO TO EXIT1;
      GO TO EXIT3;
                  END;
/* X = Y */
              IF SINE < 80H THEN DO;
     ZE=X(2);          GO TO EXIT1;       END;
              ZZ=0;
                  ZE    = 0;
             GO TO RET;
                  END;
   IF Y2 > R2 THEN DO;
/* Y > X */
            ZE    =      Y(2);
        XX=SHR(XX,DIGIT);
              IF SINE < 80H THEN GO TO EXIT1;
                          GO TO EXIT2;
                  END;
/* X > Y */
            ZE    =      X(2);
        YQ=SHR(YQ,DIGIT);
              IF SINE < 80H THEN GO TO EXIT1;
      GO TO EXIT3;
   EXIT1:
    ZZ=XX+YQ;
    IF CARRY THEN DO;
        ZZ=SCR(ZZ,1);
        ZE=ZE  +1;
   END;
         GO TO RET;
       EXIT2:
             ZZ=YQ-XX;
             CALL ADJUST;
       GO TO RET;
    EXIT3:
```

```
               ZZ=XX-YQ;
               CALL ADJUST;
  RET:  P=HIGH(ZZ);P(1)= LOW(ZZ);P(2)=ZE;
       RETURN;
END ADD;

 SUB: PROCEDURE (AD,BD,CD);
 DECLARE (AD,BD,CD) ADDRESS,
  A BASED AD BYTE,
  B BASED BD BYTE,
  C BASED CD BYTE;
 DECLARE BC (3) BYTE;
  BC(0)= B(0);
  BC(1)= B(1);
  BC(2)= B(2) XOR 80H;
 CALL ADD (AD,.BC,CD);
 END SUB;

  /* FLOATING POINT MULTIPLY ROUTINE */
     MULT: PROCEDURE(XA,YA,PA);
     DECLARE SAVE BYTE;
      DECLARE  I BYTE,
       XA ADDRESS, X BASED XA BYTE,
       YA ADDRESS, Y BASED YA BYTE,
       PA ADDRESS,P BASED PA BYTE.
      TQ ADDRESS, TT ADDRESS;
     SAVE=X(1);
     IF (X=0) OR (Y=0) THEN DO;
      ZZ=0; ZE=0: GO TO RET; END;
      X(1)=SHR(X(1),1);
      TT= SHL(DOUBLE(X),7) OR X(1);
      TQ=SHL(DOUBLE(Y),8) OR Y(1);
      ZZ=0;
      DO I=0 TO 15 BY 1;
      ZZ= SCR (ZZ,1);
      TQ= SCR(TQ,1);
      IF (CARRY) THEN DO;
       ZZ= ZZ+TT;
      END;
      END;
      IF (ZZ<8000H) THEN DO;
       ZZ= SHL(ZZ,1);
      ZE=-1;
     END;
       ELSE ZE=0;
      /* ADD EXPONENTS   */
       ZE=((X(2) AND 7FH) +(Y(2) AND 7FH)-64 +ZE)
       OR ((X(2) AND 80H)
       XOR ( Y(2) AND 80H));
  RET:  P=HIGH(ZZ);P(1)= LOW(ZZ);P(2)=ZE;
     X(1)=SAVE;
         RETURN;
      END MULT;

  /* ROUTINE TO COMPARE TWO FLOATING POINT VARIABLES */
     /*     IF X<Y COMPARE=0
            X=Y COMPARE=1
          X>Y COMPARE=2     */
COMPARE: PROCEDURE (XA,YA) BYTE;
     DECLARE X BASED XA BYTE,
             Y BASED YA BYTE,
             (XA,YA,CHECK1,CHECK2) ADDRESS,
             (XP,YP) BYTE;

  /* EQUAL EXPONENTS */
       XE=X(2) AND 80H;
       YE=Y(2) AND 80H;
         IF ( XE=YE ) THEN DO;
             XP=X(2) AND 7FH;
             YP=Y(2) AND 7FH;
          CHECK1=SHL(DOUBLE(X),8) OR X(1);
          CHECK2=SHL(DOUBLE(Y),8) OR Y(1);
```

```
             IF ( XE=80H ) THEN DO;
                IF ( XP<YP ) THEN RETURN 2;
                 IF (XP > YP ) THEN RETURN 0;
                   IF CHECK2 < CHECK1 THEN RETURN 0;
                   IF CHECK2 > CHECK1 THEN RETURN 2;
                   RETURN 1;
             END;
             IF ( XP < YP ) THEN RETURN 0;
               IF ( XP > YP ) THEN RETURN 2;
          IF CHECK2 < CHECK1 THEN RETURN 2;
          IF CHECK2 > CHECK1 THEN RETURN 0;
          RETURN 1;
          END;
           IF ( XE = 0 ) THEN RETURN 2;
            RETURN 0;
END CCMPARE;
DIV:  PROCEDURE (XA,YA,PA);
      DECLARE (XA,YA,ZA,B,XX,YQ) ADDRESS,
       PA ADDRESS,P BASED PA BYTE,
              I BYTE,
              X BASED XA BYTE,
              Y BASED YA BYTE,
            C (3) BYTE;



      XX=SHL(DOUBLE(X),8) OR X(1);
      YQ=SHL(DOUBLE(Y),8) OR Y(1);
       IF XX =0H THEN DO;
         ZZ=0H; ZE=0H; GO TO RET;
       END;
      XX=SHR(XX,1);
      YQ=SHR(YQ,1);
 ENT0: ZZ=0;
      C(2)=     X(2);
          I=1;
      B=XX-YQ;
      IF NOT CARRY THEN DO;
                  IF B=0 THEN DO;
                    ZZ=8000H;
                 ZE=((( X(2) AND 80H) XOR ( Y(2) AND 80H))
                     OR ((C(2) AND 7FH)-(Y(2) AND 7FH)+65));
                     GO TO RET;
                   END;
          GOTO ENT2;
          END;
      XX=SHL(XX,1);
      C(2)=     (C(2) AND 7FH) - 1;
 ENT1: B=XX-YQ;
      IF CARRY THEN DO;
          XX=SHL(XX,1);
          GOTO ENT3;
          END;
  /* NO CARRY, ADD ONE TO DIVIDEND TO MANTISSA AND
                                    SHIFT LEFT */
 ENT2: XX=SHL(B,1);
      ZZ=ZZ+1;
  /* CARRY SHIFT DIVIDEND MANTISSA LEFT */
 ENT3: ZZ=SHL(ZZ,1);
      I=I+1;
      IF I< 16 THEN GOTO ENT1;
  /* SET EXPONENT */
        ZE=(((    X(2) AND 80H) XOR (    Y(2) AND 80H)) OR
                     ((C(2) AND 7FH) - ( Y(2) AND 7FH)+65));
  RET:  P=HIGH(ZZ);P(1)= LOW(ZZ);P(2)=ZE;
        RETURN;
END DIV;
     DECLARE RPT BYTE;
     DECLARE (DTT,DRT) (3) BYTE;
     DECLARE (IJ,IZ,ZI) BYTE;
     DECLARE BUFF(27) BYTE;
     DECLARE (INB,OUTB) BYTE;
```

```
DECLARE FT(12) BYTE;  /* FLOATING POINT TIME ARRAY */
DECLARE (RO,RO1,ROO1,SRO,SRO1) (12) BYTE;
/* INTERMEDIATE RESULTS OF THE PROCEDURE  POLIN */
DECLARE CONS(27) BYTE;  /* CONSTANTS OF POLINOMIAL */
/* COEFFICIENTS OF THE POLINOMIALS */
DECLARE (DTTP,DRTP) (12) BYTE;
DECLARE AUX(21) BYTE;
DECLARE (EXVAL,EXDTT,EXDRT) (3) BYTE;
DECLARE RTM BYTE;
FTAB:   PROCEDURE;
        /* FILLS THE VECTOR OF FLOATING POINT TIME */
        /* NECESSARY FOR THE POLIN PROCEDURE */
        DECLARE (I,J,K) BYTE, TIM ADDRESS;
        DECLARE CTE(3) BYTE;
        CTE=0A0H:
        CTE(1)=0H;
        CTE(2)=44H;
        DO I=0 TO 3;
        K=SHL(I,4);
        J=I+I+I;
        TIM=TABLE(K);
        CALL FLOAT(.TIM,.FT(J));
        CALL DIV(.FT(J),.CTE,.FT(J));
        END;
        RETURN;
        END FTAB:
POLIN:  PROCEDURE;
        /* THIS PROCEDURE COMPUTES THE CONSTANTS
           AND THE COEFFICIENTS
           NECESSARY FOR THE EXTRAPOLATION BY
           ORTHOGONAL POLINOMIALS. THE CONSTANTS
           WILL BE STORED IN THE ARRAY CONS AND
           THE COEFFICIENTS IN ARRAYS DTTP AND DRTP.
           THE SYMBOLS GIVEN IN THE COMMENTS
           REFER TO THE FORTRAN SIMULATION,I.E.
           SO,BO,S1,B1,C1,S2,B2,C2,S3...    */
        DECLARE (I,J,K,L) BYTE;
        DO I=1 TO 27;  CONS(I)=0;  END;
        DO I=1 TO 12; DTTP(I)=0;
        DRTP(I)=0;  END;
        DO I=1 TO 18;  AUX(I)=0;  END;
        /* SO=4.0 */
        CONS=80H;  CONS(1)=0H;  CONS(2)=43H;
        /* BO: */
        DO I=0 TO 3;
        J=I+I+I;
        CALL ADD(.CONS(3),.FT(J),.CONS(3));
        END;
        CALL DIV(.CONS(3),.CONS,.CONS(3));
        /* S1: */
        DO I=0 TO 3;
        J=I+I+I;
        CALL SUB(.FT(J),.CONS(3),.AUX);
          DO K=0 TO 2;
          RO(J+K)=AUX(K);
          END;
        CALL MULT(.AUX,.AUX,.AUX);
          DO K=0 TO 2;
          SRO(J+K)=AUX(K);
          END;
        CALL ADD(.CONS(6),.AUX,.CONS(6));
        END;
        /* B1,C1: */
        DO I=0 TO 3;
        J=I+I+I;
        CALL MULT(.FT(J),.SRO(J),.AUX);
        CALL ADD(.CONS(9),.AUX,.CONS(9));
        END;
        CALL DIV(.CONS(9),.CONS(6),.CONS(9));
        CALL DIV(.CONS(6),.CONS,.CONS(12));
```

```
/* S2: */
DO I=0 TO 3;
J=I+I+I;
CALL SUB(.FT(J),.CONS(9),.AUX);
CALL MULT(.AUX,.RO(J),.AUX);
CALL SUB(.AUX,.CONS(12),.AUX);
   DO K=0 TO 2;
   RO1(J+K)=AUX(K);
   END;
CALL MULT(.AUX,.AUX,.AUX);
   DO K=0 TO 2;
   SRO1(J+K)=AUX(K);
   END;
CALL ADD(.CONS(15),.AUX,.CCNS(15));
END;
/* B2,C2: */
DO I=0 TO 3;
J=I+I+I;
CALL MULT(.FT(J),.SRO1(J),.AUX);
CALL ADD(.CONS(18),.AUX,.CCNS(18));
END;
CALL DIV(.CONS(18),.CONS(15),.CONS(18));
CALL DIV(.CONS(15),.CONS(6) ,.CONS(21));
/* S3: */
DO I=0 TO 3;
J=I+I+I;
CALL SUB(.FT(J),.CONS(18),.AUX);
CALL MULT(.AUX,.RO1(J)..AUX(3));
CALL MULT(.CONS(21),.RO(J),.AUX);
CALL SUB(.AUX(3),.AUX,.AUX);
   DO K=C TO 2;
   ROO1(J+K)=AUX(K);
   END;
CALL MULT(.AUX,.AUX,.AUX);
CALL ADD(.CONS(24),.AUX,.CCNS(24));
END;
/* C(1),D(1): */
DO I=0 TO 3;
J=10+SHL(I,4);
K=J+3;
CALL ADD(.DTTP,.TABLE(J),.DTTP);
CALL ADD(.DRTP,.TABLE(K),.DRTP);
END;
CALL DIV(.DTTP,.CONS,.DTTP);
CALL DIV(.DRTP,.CONS,.DRTP);
/* C(2),D(2): */
DO I=0 TO 3;
J=10+SHL(I,4);
K=J+3;
L=I+I+I;
CALL MULT(.TABLE(J),.RO(L),.AUX);
CALL ADD(.DTTP(3),.AUX,.DTTP(3));
CALL MULT(.TABLE(K),.RO(L),.AUX);
CALL ADD(.DRTP(3),.AUX,.DRTP(3));
END;
CALL DIV(.DTTP(3),.CONS(6),.DTTP(3));
CALL DIV(.DRTP(3),.CONS(6),.DRTP(3));
/* C(3),D(3): */
DO I=0 TO 3;
J=10+SHL(I,4);;
K=J+3;
L=I+I+I;
CALL MULT(.TABLE(J),.RO1(L),.AUX);
CALL ADD(.DTTP(6),.AUX,.DTTP(6));
CALL MULT(.TABLE(K),.RO1(L),.AUX);
CALL ADD(.DRTP(6),.AUX,.DRTP(6));
END;
CALL DIV(.DTTP(6),.CONS(15),.DTTP(6));
CALL DIV(.DRTP(6),.CONS(15),.DRTP(6));
/* C(4),D(4): */
DO I=0 TO 3;
J=10+SHL(I,4);
```

```
         K=J+3;
         L=I+I+I;
         CALL MULT(.TABLE(J),.R001(L),.AUX);
         CALL ADD(.DTTP(9),.AUX,.DTTP(9));
         CALL MULT(.TABLE(K),.R001(L),.AUX);
         CALL ADD(.DRTP(9),.AUX,.DRTP(9));
         END;
         CALL DIV(.DTTP(9),.CONS(24),.DTTP(9));
         CALL DIV(.DRTP(9),.CONS(24),.DRTP(9));
         RETURN;
         END POLIN;
EXTRAP:  PROCEDURE(TM,AVEC);
         /* EXTRAPOLATES VALUES FOR DOWN RANGE TRAVEL
            AND DISTANCE TO TARGET, GIVEN THE
            VARIABLE TIME  */
         DECLARE (TIM,AVEC) ADDRESS, FT(3) BYTE;
         DECLARE (TM,I) BYTE, VEC BASED AVEC BYTE;
         DECLARE CTE(3) BYTE INITIAL(0A0H,0H,44H);
         DO I=0 TO 21;
         AUX(I)=0;
         END;
         TIM=TM;
         CALL FLOAT(.TIM,.FT);
         CALL DIV(.FT,.CTE,.FT);
         CALL SUB(.FT,.CONS(3),.AUX);
         CALL MULT(.AUX,.VEC(3),.AUX(3));
         CALL SUB(.FT,.CONS(9),.AUX(6));
         CALL MULT(.AUX(6),.AUX,.AUX(6));
         CALL SUB(.AUX(6),.CONS(12),.AUX(6));
         CALL SUB(.FT,.CONS(18),.AUX(9));
         CALL MULT(.AUX(9),.AUX(6),.AUX(9));
         CALL MULT(.CONS(21),.AUX,.AUX(18));
         CALL SUB(.AUX(9),.AUX(18),.AUX(9)):
         CALL ADD(.VEC,.AUX(3),.AUX(12));
         CALL MULT(.VEC(6),.AUX(6),.AUX(15));
         CALL ADD(.AUX(12),.AUX(15),.AUX(12));
         CALL MULT(.VEC(9),.AUX(9),.AUX(15));
         CALL ADD(.AUX(12),.AUX(15),.AUX(12));
         DO I=0 TO 2;
         EXVAL(I)=AUX(12+I);
         END;
         RETURN;
         END EXTRAP;
COMP:    PROCEDURE(DT) BYTE;
         /* AUXILIARY PROCEDURE FOR THE SEARCH:
            PERFORMS A STEPWISE COMPARISON BETWEEN
            THE CURVES OF DOWN RANGE TRAVEL AND
            DISTANCE TO TARGET  */
         /* IF THE PROCEDURE RETURNS 0,THERE IS NO
            SOLUTION.  IF IT RETURNS  1, THE
            INTERSECTION HAS BEEN PASSED DURING THE
            SEARCH */
         DECLARE (I,J,DT,CP) BYTE;
         DO I=1 TO 10;
         RTM=RTM+DT;
         CALL EXTRAP(RTM,.DTTP);
         DO J=0 TO 2;
         EXDTT(J)=EXVAL(J);
         END;
         CALL EXTRAP(RTM,.DRTP);
         DO J=0 TO 2;
         EXDRT(J)=EXVAL(J);
         END;
         IF (DT AND 80H) =  0 THEN DO;
           CP=COMPARE(.EXDRT,.EXDTT);
           IF ((CP=2) OR (CP=1)) THEN RETURN 1;  END;
           ELSE DO;
           CP=COMPARE(.EXDTT,.EXDRT);
           IF ((CP=2) OR (CP=1)) THEN RETURN 1;  END;
         END;
         RETURN 0;
         END COMP;
```

```
ROUND:   PROCEDURE BYTE;
         /* TO CHOOSE THE CLOSEST TIME CORRESPONDING
            TO THE INTERSECTION OF THE CURVES  */
         DECLARE (DIFO,DIFT) (3) BYTE, J BYTE;
         CALL EXTRAP(RTM,.DTTP);
         DO J=0 TO 2;
         EXDTT(J)=EXVAL(J);
         END;
         CALL EXTRAP(RTM,.DRTP);
         DO J=0 TO 2;
         EXDRT(J)=EXVAL(J);
         END;
         CALL SUB(.EXDRT,.EXDTT,.DIFO);
         RTM=RTM-1;
         CALL EXTRAP (RTM,.DTTP);
         DO J=0 TO 2;
         EXDTT(J)=EXVAL(J);
         END;
         CALL EXTRAP(RTM,.DRTP);
         DO J=0 TO 2;
         EXDRT(J)=EXVAL(J);
         END;
         CALL SUB(.EXDTT,.EXDRT,.DIFT);
         IF COMPARE(.DIFO,.DIFT)=2 THEN RETURN RTM;
         ELSE RETURN RTM+1;
         END ROUND;
SEARCH:  PROCEDURE BYTE;
         /* THIS PROCEDURE GOVERNS THE SEARCH FOR
            THE INTERSECTION. IT WILL RETURN THE
            TIME OF INTERSECTION ,IN DECIMALS OF
            SECOND. THE MEAN ERROR, BECAUSE OF THE
            PROCEDURE ROUND, WILL BE 0.025 SECONDS  */
         DECLARE (RESUL,RDRTM) BYTE;
         IF COMP(20)=0 THEN RETURN 0; /*  NO SOLUTION */
         RESUL=COMP(-5);
         RESUL=COMP(1);
         RDRTM=ROUND;
         RETURN RDRTM;
         END SEARCH;


SENDBUFF: PROCEDURE;
         DECLARE  IX BYTE;
         IZ=SHL(INB,3)+INB;   /* MULT BY 9 */
         INB=INB+1;
         TI=TI+1;
         DO IX=0 TO 8;
         BUFF(IZ+IX)=TABLE(TI+IX);
         END;
         RETURN;
         END SENDBUFF;


REALT:   PROCEDURE;
         /* DISPLAYS THE REAL TIME */
         DECLARE (RTIME,RST) BYTE;
         RTIME=CLOCK;
         CALL CRLF;
         CALL PRINT(.'THE TIME NOW IS: $');
         RST=INTEG(RTIME,2);
         CALL DECIM(RST);
         CALL PRINT(.' SECONDS$');
         CALL CRLF;
         RETURN;
         END REALT;


RELPT:   PROCEDURE;
         /* COMPUTES AND DISPLAYS THE RELEASE POINT */
         DECLARE (RST,J) BYTE;
         RTM=80;
         RPT=SEARCH;
```

```
           IF RPT=0 THEN CALL PRINT(.'NO SOLUTION$');
           ELSE DO;
           CALL CRLF;
           CALL REALT;
           CALL CRLF;
           CALL PRINT(.'RELEASE POINT:$');
           CALL CRLF;
           CALL PRINT(.'TIME=$');
           RST=INTEG(RPT,2);
           CALL DECIM(RST);
           CALL PRINT(.' SECONDS$');
           CALL CRLF;
           CALL EXTRAP(RPT,.DTTP);
           DO J=0 TO 2;
             DTT(J)=EXVAL(J);
           END;
           CALL EXTRAP(RPT,.DRTP);
           DO J=0 TO 2;
             DRT(J)=EXVAL(J);
           END;
           CALL PRINT(.'DISTANCE TO TARGET=$');
           CALL DSPLV(.DTT,4);
           CALL PRINT(.' YARDS$');
           CALL CRLF;
           CALL PRINT(.'DOWN RANGE TRAVEL=$');
           CALL DSPLV(.DRT,4);
           CALL PRINT(.' YARDS$');
           CALL CRLF;
           END;
           RETURN;
           END RELPT;


   INTSERV:  PROCEDURE INTERRUPT 3;
           DECLARE (I,J) BYTE;
           DISABLE;
           OUTPUT(3)=INTER;
           IDLE=1;
           J=SHL(NUMB,4)+13;
           NUMB=NUMB+1;
           DO I=0 TO 2;
           OUTPUT(3)=I+1;
           CALL TIME(3);
           /* TO GIVE TIME TO THE OTHER COMPUTER */
           TABLE(J+I)=INPUT(7);
           END;
           IF INB > OUTB THEN DO;
           ZI=SHL(OUTB,3)+OUTB;
           OUTB=OUTB+1;
           IDLE=0;
           OUTPUT(3)=FROM$BUFFER;
           DO WHILE INPUT(7) <> FROM$BUFFER;  END;
           /* WAITS FOR BAL. TO BE READY */
           DO I=0 TO 8;
             DO WHILE INPUT(7) <> I+1;  END;
             OUTPUT(3)=BUFF(ZI+I);
           END;
           END;
           ELSE OUTPUT(3)=EMPTY;
           ENABLE;
           RETURN;
           END INTSERV;


DO FOREVER;
   ENABLE;
   IDLE=1;
   INDX=13;
   INB=0;
   OUTB=0;
   NUMB=0;
   OUTPUT(3)=0;
```

```
   DO WHILE INPUT(7) <> INIT;  END;
   /* WAITS FOR  PROMPT SIGNAL FROM BAL.  */
/* GO ON UNTIL CONTROL-C IS TYPED. */
   CALL CRLF;
   CALL PRINT (.'BEGINING OF A RUN: $');
       CALL CRLF;
   CALL PRINT(.'THE PROGRAM HAS A TABLE OF AIRCRAFT$');
   CALL PRINT(.' POSITIONS FOR 4 DIFFERENT RUNS,$');
   CALL CRLF;
   CALL PRINT(.'USING THE BOMB MK82 UNRETARDED$');
   CALL PRINT(.',IDNO=10.$');
   CALL CRLF;
   CALL PRINT(.'ENTER A RUN NUMBER FROM 1 TO 4.$');
   CALL CRLF;
   CALL CRLF;
   CALL PRINT(.'RUN=$');
   CALL READ;
   IF RDBUFF(2)=03H THEN GO TO OH;
   RUN=CONV(RDBUFF(2));
   CALL CRLF;
   CALL CRLF;
   CALL PRINT(.'INPUT WEAPON ID NUMBER AFTER    ''=''  $');
   CALL CRLF;
   CALL PRINT(.'IDNO HAS TO BE FROM 1 TO 28. $');
   CALL CRLF;
   CALL PRINT(.'IF YOU WANT TO QUIT ,TYPE CONTROL-C. $');
   CALL CRLF;
   CALL CRLF;
   CALL PRINT(.'IDNO= $');
   CALL READ;
   IF RDBUFF(2)=03H THEN GO TO OH;
   IF RDBUFF(1)=1 THEN
     DO;
       RDBUFF(3)=RDBUFF(2);
       RDBUFF(2)=0;
     END;
   IDNO=10*CONV(RDBUFF(2))+CONV(RDBUFF(3));
   CALL CRLF;
   CALL CRLF;
   CALL PRINT(.'EXECUTION WILL BEGIN WHEN CLOCK=0$');
   CALL CRLF;
   OUTPUT(3)=INIT;   /* TELLS BAL. THAT THE IDNO WILL
                        BE SENT  */
   DO WHILE INPUT(7) <> SEND$IDNO;  END;
   /*  WAITS FOR THE PROMPT  SIGNAL  */
   OUTPUT(3)=IDNO;  /* SEND WEAPON ID NUMBER */
   RID=(RUN-1)*520;
   DO WHILE CLOCK <> 0;   /* WAIT UNTIL CLOCK COUNT= 0 */
   END;
   DO I=0 TO 3;
      TI=16*I;
      NI=13+26*I;
       DO J=0 TO 12;
       TABLE(TI+J)=NAVTAB(NI+J+RID);
       /* SIMULATES THE RECEIVING OF DATA FROM NAV.  */
       END;
   DISABLE;
   IF IDLE THEN DO;
   IDLE=0;
   DO WHILE INPUT(7) <> DAT;  END;
   OUTPUT(3)=DAT;
   TI=TI+1;
   DO IJ=0 TO 8;
     DO WHILE INPUT(7) <> IJ+1;  END;
     OUTPUT(3)=TABLE(TI+IJ);
   END;
   END;  /*  IF  */
   ELSE CALL SENDBUFF;
   ENABLE;
   /* TO SYNCHRONIZE EXECUTION WITH THE TABLE OF DATA */
   J=SHL(I,1)+1;
   DO WHILE CLOCK < ( SHL(J,3)+SHL(J,1) );  END;
```

```
    DISABLE;
    INDX=INDEX(CLOCK)*13+RID;
    CALL DISPLAY(INDX);
    ENABLE;
    END;    /*  DO I=0 TO 3 */
    DO WHILE NOT IDLE;
    /*  THIS FLAG IS SET BY INTSERV */
    END;  /* WHILE */
    CALL CRLF;
    CALL PRINT(.'START TO COMPUTE RELEASE POINT$');
    CALL CRLF;
    CALL REALT;
    CALL CRLF;
    CALL CRLF;
    CALL FTAB;
    CALL POLIN;
    /* EXTRAPOLATE DOWNRANGE AND DIST. TO TARGET
       CURVES AND FIND THE INTERSECTION  */
    CALL RELPT;
    /* COMPUTES AND DISPLAYS THE RELEASE POINT */
    CALL CRLF;
    CALL PRINT(.'FROM THE TABLE OF DATA:$');
    /*  DISPLAYS DATA  FROM TABLE AT TIME OF RELEASE    */
    /*  TO CHECK THE RESULT OF EXTRAPOLATION  */
    CALL CRLF;
    INDX=INDEX(RPT)*13+RID;
    CALL DISPLAY(INDX);
    CALL CRLF;
    DO WHILE INPUT(7)<>DAT;  END;
    OUTPUT(3)=DAT;
    IDLE=0;
    DO IJ=0 TO 8;
      DO WHILE INPUT(7) <> IJ+1;  END;
      OUTPUT(3)=NAVTAB(INDX+1+IJ);
    END;
    CALL PRINT(.'FOR THIS POSITION:$');
    CALL CRLF;
    DO WHILE NOT IDLE;  END;
    CALL PRINT(.'DOWN RANGE TRAVEL=$');
    CALL DSPLV(.TABLE(77),4);
    CALL PRINT(.' YARDS$');
    CALL CRLF;
    CALL CRLF;
 END; /*  FOREVER      */
    EOF
```

```
     /***************************************************
     *                                                  *
     *   BALLISTICS' COMPUTER PROGRAM                   *
     *                                                  *
     ***************************************************/
     DECLARE FOREVER LITERALLY 'WHILE 1';
     DECLARE INIT LITERALLY '0C0H';
     DECLARE DAT  LITERALLY '0D0H';
     DECLARE SEND$IDNO   LITERALLY '0ABH';
     DECLARE FROM$BUFFER LITERALLY '0BBH';
     DECLARE EMPTY LITERALLY '0CCH';
     DECLARE INTER LITERALLY '0B0H';
     DECLARE (IDNO,COUNT) BYTE;
     DECLARE INDAT(9) BYTE;
     DECLARE X(3) BYTE;


     SERV$ONE: PROCEDURE;
             /* TO  RECEIVE WEAPON ID NUMBER */
             OUTPUT(7)=SEND$IDNO;
             CALL TIME(3);
             IDNO=INPUT(7);
             RETURN;
             END SERV$ONE;


     SERV$TWO: PROCEDURE;
             /* WILL RECEIVE SPEED,ALTITUDE AND DIVE ANG. */
             DECLARE (I,IJ) BYTE;
     LOCP:   DO IJ=0 TO 8;
             OUTPUT(7)=IJ+1;
             CALL TIME(3);
             INDAT(IJ)=INPUT(7);
             END;
     /* NOW THE DOWN RANGE TRAVEL WILL BE COMPUTED */
 DO;
 DECLARE ZE BYTE, ZZ ADDRESS;
DECLARE YE BYTE,XE BYTE;
  /* FLOATING POINT ADD ROUTINE*/
ADD: PROCEDURE (XA,YA,PA);
     DECLARE (XA,YA,ZA) ADDRESS,
       PA ADDRESS, P BASED PA BYTE,
             (I,R2,Y2,DIGIT,SINE) BYTE,
             X BASED XA BYTE,
             Y BASED YA BYTE,
              (XX,YQ) ADDRESS;

  /* PROCEDURE TO LEFT JUSTIFY MANTISSA IN BINARY */
    ADJUST: PROCEDURE;
      DO I=0 TO 15;
         IF (ZZ AND 8000H)=8000H THEN RETURN;
        ZZ=SHL(ZZ,1);
        ZE=ZE-1;
     END;   RETURN;
      END ADJUST;

     XX=SHL(DOUBLE(X),8) OR X(1);
     YQ=SHL(DOUBLE(Y),8) OR Y(1);
     R2=      X(2) AND 7FH;
     Y2=      Y(2) AND 7FH;
     DIGIT=R2-Y2;
      SINE =(X(2) AND 80H) XOR (Y(2) AND 80H);
      IF (DIGIT AND 80H)<>0    THEN DIGIT=-DIGIT;
     IF DIGIT >= 16  THEN DO;
  /* VARIABLES NOT WITHIN SIGNIFICANCE RANGE */
      IF R2 > Y2 THEN DO;
       ZZ=XX; ZE=X(2); GO TO RET;
        END;
       ZZ=YQ; ZE=Y(2); GO TO RET;
        END;
      IF Y2 = R2 THEN DO;
  /* EXPONENTS EQUAL IN ABSOLUTE VALUE     */
```

```
            IF YQ>XX THEN DO;
  /* Y > X */
               ZE= Y(2);
                IF SINE < 80H THEN GO TO EXIT1;
                            GO TO EXIT2;
                     END;
           IF YQ<XX THEN DO;
  /* X > Y */
                     ZE =X(2);
                IF SINE < 80H THEN GO TO EXIT1;
         GO TO EXIT3;
                     END;
  /* X = Y */
                IF SINE < 80H THEN DO;
        ZE=X(2);      GO TO EXIT1;        END;
                ZZ=0;
                    ZE   = 0;
                GO TO RET;
                     END;
     IF Y2 > R2 THEN DO;
  /* Y > X */
               ZE    =      Y(2);
          XX=SHR(XX,DIGIT);
                IF SINE < 80H THEN GO TO EXIT1;
                            GO TO EXIT2;
                     END;
  /* X > Y */
               ZE    =      X(2);
          YQ=SHR(YQ,DIGIT);
                IF SINE < 80H THEN GO TO EXIT1;
         GO TO EXIT3;
     EXIT1:
      ZZ=XX+YQ;
      IF CARRY THEN DO;
          ZZ=SCR(ZZ,1);
          ZE=ZE  +1;
     END;
           GO TO RET;
         EXIT2:
                ZZ=YQ-XX;
                CALL ADJUST;
         GO TO RET;
      EXIT3:
                ZZ=XX-YQ;
                CALL ADJUST;
  RET:   P=HIGH(ZZ);P(1)= LOW(ZZ);P(2)=ZE;
        RETURN;
END ADD;

 SUB: PROCEDURE (AD,BD,CD);
 DECLARE (AD,BD,CD) ADDRESS,
  A BASED AD BYTE,
  B BASED BD BYTE,
  C BASED CD BYTE;
 DECLARE BC (3) BYTE;
  BC(0)= B(0);
  BC(1)= B(1);
  BC(2)= B(2) XOR 80H;
 CALL ADD (AD,.BC,CD);
 END SUB;

  /* FLOATING POINT MULTIPLY ROUTINE */
     MULT: PROCEDURE(XA,YA,PA);
     DECLARE   SAVE BYTE;
      DECLARE I BYTE,
       XA ADDRESS, X BASED XA BYTE,
       YA ADDRESS, Y BASED YA BYTE,
       PA ADDRESS,P BASED PA BYTE,
      TQ ADDRESS, TT ADDRESS;
     SAVE=X(1);
     IF (X=0) OR (Y=0) THEN DO;
      ZZ=0; ZE=0; GO TO RET; END;
```

```
      X(1)= SHR(X(1),1);
      TT = SHL(DOUBLE(X),7) OR X(1);
      TQ=SHL(DOUBLE(Y),8) OR Y(1);
     ZZ=0;
     DO I=0 TO 15 BY 1;
      ZZ= SCR(ZZ,1);
      TQ= SCR(TQ,1);
      IF (CARRY) THEN DO;
        ZZ = ZZ+TT;
      END;
     END;
     IF (ZZ<8000H) THEN DO;
       ZZ= SHL(ZZ,1);
      ZE=-1;
     END;
       ELSE ZE=0;
      /* ADD EXPONENTS   */
       ZE=((X(2) AND 7FH) +(Y(2) AND 7FH)-64 +ZE)
      OR ((X(2) AND 80H)
      XOR ( Y(2) AND 80H));
  RET:  P=HIGH(ZZ);P(1)= LOW(ZZ);P(2)=ZE;
     X(1)=SAVE;
         RETURN;
      END MULT;

  /* ROUTINE TO COMPARE TWO FLOATING POINT VARIABLES */
     /*     IF X<Y COMPARE=0
           X=Y COMPARE=1
         X>Y COMPARE=2     */
COMPARE: PROCEDURE (XA,YA) BYTE;
     DECLARE X BASED XA BYTE,
             Y BASED YA BYTE,
             (XA,YA,CHECK1,CHECK2) ADDRESS,
             (XP,YP) BYTE;

  /* EQUAL EXPONENTS */
       XE=X(2) AND 80H;
       YE=Y(2) AND 80H;
         IF ( XE=YE ) THEN DO;
             XP=X(2) AND 7FH;
             YP=Y(2) AND 7FH;
          CHECK1=SHL(DOUBLE(X),8) OR X(1);
          CHECK2=SHL(DOUBLE(Y),8) OR Y(1);
             IF ( XE=80H ) THEN DO;
                IF ( XP<YP ) THEN RETURN 2;
                 IF (XP > YP ) THEN RETURN 0;
                   IF CHECK2 < CHECK1 THEN RETURN 0;
                   IF CHECK2 > CHECK1 THEN RETURN 2;
                   RETURN 1;
             END;
             IF ( XP < YP ) THEN RETURN 0;
               IF ( XP > YP ) THEN RETURN 2;
          IF CHECK2 < CHECK1 THEN RETURN 2;
          IF CHECK2 > CHECK1 THEN RETURN 0;
          RETURN 1;
          END;
           IF ( XE = 0 ) THEN RETURN 2;
            RETURN 0;
END COMPARE;
DIV:  PROCEDURE (XA,YA,PA);
      DECLARE (XA,YA,ZA,B,XX,YQ) ADDRESS,
       PA ADDRESS,P BASED PA BYTE,
              I BYTE,
              X BASED XA BYTE;
              Y BASED YA BYTE,
            C (3) BYTE;


      XX=SHL(DOUBLE(X),8) OR X(1);
      YQ=SHL(DOUBLE(Y),8) OR Y(1);
       IF XX =0H THEN DO;
```

```
         ZZ=0H; ZE=0H; GO TO RET;
       END;
      XX=SHR(XX,1);
      YQ=SHR(YQ,1);
 ENT0: ZZ=0;
      C(2)=     X(2);
          I=1;
      B=XX-YQ;
      IF NOT CARRY THEN DO;
                   IF B=0 THEN DO;
                     ZZ=8000H;
                  ZE=((( X(2) AND 80H) XOR (Y(2) AND 80H))
                      OR ((C(2) AND 7FH)-(Y(2) AND 7FH)+65));
                      GO TO RET;
                    END;
          GOTO ENT2;
          END;
      XX=SHL(XX,1);
      C(2)=     (C(2) AND 7FH) - 1;
 ENT1: B=XX-YQ;
      IF CARRY THEN DO;
          XX=SHL(XX,1);
          GOTO ENT3;
          END;
  /* NO CARRY, ADD ONE TO DIVIDEND TO MANTISSA AND
                                   SHIFT LEFT */
 ENT2: XX=SHL(B,1);
      ZZ=ZZ+1;
  /* CARRY SHIFT DIVIDEND MANTISSA LEFT */
 ENT3: ZZ=SHL(ZZ,1);
      I=I+1;
      IF I< 16 THEN GOTO ENT1;
  /* SET EXPONENT */
        ZE=(((     X(2) AND 80H) XOR (     Y(2) AND 80H)) OR
                  ((C(2) AND 7FH) - (    Y(2) AND 7FH)+65));
  RET:  P=HIGH(ZZ);P(1)= LOW(ZZ);P(2)=ZE;
        RETURN;
END DIV;
      SQRT:  PROCEDURE (XA,PA);
      /*  ASSUME THAT XA IS A POSITIVE REAL NUMBER  */
       DECLARE XA ADDRESS, X BASED XA BYTE;
       DECLARE PA ADDRESS, P BASED PA BYTE;
       DECLARE (C,C1,C2,C3,B,B1,B2) BYTE;
      /* INITIAL APPROXIMATION FOR THE ROOT IS
               MANT * EXP/2  */
       DECLARE R BYTE;
       B=X;  B1= X(1);
        B2=X(2)-64; IF (B2 AND 80H) =0 THEN B2=SHR(B2,1)+64;
                       ELSE DO; B2=-B2; B2=SHR(B2,1);
                       B2=64-B2;
                             END;
      DO R= 1 TO 4;
            CALL DIV(XA,.B,.C);
           CALL ADD(.C,.B,.B);
            B(2)=B(2)-1;
       END;
      P=B; P(1)=B(1);  P(2)=B(2);
         RETURN;
      END SQRT;

       DECLARE (T,T1,T2,Z,Z1,Z2 )BYTE;
DECLARE       (ID,TEMP,TEMP1,TEMP2,ENTRY,ENTRY1,ENTRY2)BYTE;
    FUNCTION: PROCEDURE(I,XA);
  /*  I=0      CDTABLE
      I=1      ATMOSPHERE
      I=2      COSINE
      I=3      ATAN1
      I=4 ATAN2     */
     DECLARE ETWO DATA
      ( 0H, 0H, 0H, 0C9H, 0EH, 3BH, 0C9H, 0EH, 3CH, 96H,
      0CAH, 3DH, 0C9H, 0EH, 3DH, 0FBH, 51H, 3DH, 96H, 0CAH,
      3EH, 0AFH, 0ECH, 3EH, 0C9H, 0EH, 3EH, 0E2H, 2FH, 3EH,
```

```
      0FBH, 51H, 3EH, 8AH, 39H, 3FH, 96H, 0CAH, 3FH, 0A3H,
      5BH, 3FH, 0AFH, 0ECH, 3FH, 0BCH, 7DH, 3FH, 0C9H, 0EH,
      3FH, 0D5H, 9EH, 3FH, 0E2H, 2FH, 3FH, 0EEH, 0C0H, 3FH,
      0FBH, 51H, 3FH, 83H, 0F1H, 40H, 8AH, 39H, 40H, 90H,
      82H, 40H, 96H, 0CAH, 40H, 9DH, 12H, 40H, 0A3H, 5BH,
      4CH, 0A9H, 0A3H, 40H, 0AFH, 0ECH, 40H, 0B6H, 34H, 40H,
      0BCH, 7DH, 40H, 0C2H, 0C5H, 40H, 0C9H, 0EH, 40H, 0CFH,
      56H, 4CH, 0D5H, 9EH, 40H, 0DBH, 0E7H, 40H, 0E2H, 2FH,
      4CH, 0E8H, 78H, 40H, 0EEH, 0C0H, 40H, 0F5H, 09H, 40H,
      0FBH, 51H, 40H, 80H, 0CCH, 41H, 83H, 0F1H, 41H, 87H,
      15H, 41H, 8AH, 39H, 41H, 8DH, 5DH, 41H, 90H, 82H, 41H,
      93H, 0A6H, 41H, 96H, 0CAH, 41H, 99H, 0EEH, 41H, 9DH,
      12H, 41H, 0A0H, 37H, 41H, 0A3H, 5BH, 41H, 0A6H, 7FH,
      41H, 0A9H, 0A3H, 41H, 0ACH, 0C8H, 41H, 0AFH, 0ECH,
      41H, 0B3H, 10H, 41H, 0B6H, 34H, 41H, 0B9H, 58H, 41H,
      0BCH, 7DH, 41H, 0BFH, 0A1H, 41H, 0C2H, 0C5H, 41H,
      0C5H, 0E9H, 41H, 0C9H, 0EH, 41H);
     DECLARE VTWO DATA
     ( 80H, 0H, 41H, 0FFH, 0ECH, 40H, 0FFH, 0B1H, 40H, 0FFH,
       4EH, 40H, CFEH, 0C4H, 40H, 0FEH, 13H, 40H, 0FDH, 3AH,
       40H, 0FCH, 33H, 40H, 0FBH, 14H, 40H, 0F9H, 0C7H, 40H,
       0F8H, 54H, 40H, 0F6H, 0BAH, 40H, 0F4H, 0FAH, 40H,
       0F3H, 14H, 40H, 0F1H, 09H, 40H, 0EEH, 0D8H, 40H,
       0ECH, 83H, 40H, 0EAH, 0AH, 40H, 0E7H, 6CH, 40H, 0E4H,
       0AAH, 40H, 0E1H, 0C6H, 40H, 0DEH, 0BEH, 40H, 0DBH,
       94H, 40H, 0D8H, 49H, 40H, 0D4H, 0DBH, 40H, 0D1H,
      4EH, 4CH, 0CDH, 9FH, 40H, 0C9H, 0D2H, 40H, 0C5H,
       0E5H, 40H, 0C1H, 0D9H, 40H, 0BDH, 0B0H, 40H, 0B9H,
       69H, 40H, 0B5H, 06H, 40H, 0B0H, 87H, 40H, 0ABH, 0ECH,
       40H, 0A7H, 37H, 40H, 0A2H, 69H, 40H, 9DH, 81H, 40H,
       98H, 81H, 40H, 93H, 6AH, 40H, 8EH, 38H, 40H, 88H,
       0F7H, 40H, 83H, 9EH, 40H, 0FCH, 61H, 3FH, 0F1H, 5FH,
       3FH, 0E6H, 38H, 3FH, 0DAH, 0EDH, 3FH, 0CFH, 80H, 3FH,
       0C3H, 0F4H, 3FH, 0B8H, 49H, 3FH, 0ACH, 82H, 3FH,
      0A0H, 0A0H, 3FH, 94H, 0A5H, 3FH, 88H, 94H, 3FH, 0F8H,
       0DBH, 3EH, 0E0H, 68H, 3EH, 0C7H,  0D2H, 3EH, 0AFH,
       1DH, 3EH, 96H, 4DH, 3EH, 0FAH, 0CDH, 3DH, 0C8H, 0D8H,
       3DH, 96H, 0C5H, 3DH, 0C9H, 34H, 36H, 0C9H, 7FH, 3BH,
       CE9H, 51H, 32H);
     DECLARE DTWO DATA
     ( 0C9H, 05H, 0BAH, 96H, 0C1H, 0BCH, 0FBH, 27H, 0BCH,
       0AFH, 0B3H, 0BDH, 0E1H, 0B8H, 0BDH, 89H, 0CDH, 0BEH,
       0A2H, 0A8H, 0BEH, 0B8H, 6BH, 0BEH, 0D4H, 11H, 0BEH,
       0ECH, 97H, 0BEH, 82H, 7BH, 0BFH, 87H, 98H, 0BFH, 9AH,
       9EH, 0BFH, 0A6H, 8CH, 0BFH, 0B2H, 61H, 0BFH, 03EH,
       1AH, 0BFH, 0C9H, 0B6H, 0BFH, 0D5H, 33H, 0BFH, 0E0H,
       8EH, 0BFH, 0EBH, 0C8H, 0BFH, 0F6H, 0DDH, 0BFH, 80H,
       0E5H, 0C0H, 86H, 49H, 0C0H, 8BH, 98H, 0C0H, 90H,
       0D1H, 0C0H, 95H, 0F4H, 0C0H, 9BH, 00H, 0C0H, 9FH,
       0F4H, 0C0H, 0A4H, 0CFH, 0C0H, 0A9H, 91H, 0C0H, 0AEH,
       39H, 0C0H, 0B2H, 0C6H, 0C0H, 0B7H, 37H, 0C0H, 0BBH,
       8CH, 0C0H, 0BFH, 0C4H, 0C0H, 0C3H, 0DFH, 0C0H, 0C7H,
       0CBH, 0C0H, 0CBH, 0B9H, 0C0H, 0CFH, 77H, 0C0H, 0D3H,
       15H, 0C0H, 0D6H, 93H, 0C0H, 0D9H, 0EFH, 0C0H, 0DDH,
      2AH, 0C0H, 0E0H, 43H, 0C0H, 0E3H, 39H, 0C0H, 0E6H,
       0CH, 0C0H, 0E8H, 0B6H, 0C0H, 0EBH, 48H, 0C0H, 0EDH,
      0AFH, 0C0H, 0EFH, 0F2H, 0C0H, 0F2H, 10H, 0C0H, 0F4H,
       C9H, 0C0H, 0F5H, 0DCH, 0C0H, 0F7H, 89H, 0C0H, 0F9H,
       10H, 0C0H, 0FAH, 70H, 0C0H, 0FBH, 0AAH, 0C0H, 0FCH,
       0BDH, 0C0H, 0FDH, 0A9H, 0C0H, 0FEH, 6EH, 0C0H, 0FFH,
       0CH, 0C0H, 0FFH, 82H, 0C0H, 0FFH, 0D1H, CCCH, 0FFH,
       CF9H, 0C0H, 00H, 00H, 00H);
       DECLARE (E,V,D,V1,D1) (3) ADDRESS  INITIAL(
       3000H,3300H, 2D77H,
       3100H,33B7H,2E3AH,
       3200H,346EH,2EFDH,
0000H,3525H,0C00H,
       0000H,35DCH, 0000H),
          I BYTE,
      LOOK (3) BYTE INITIAL (252,181,192),
     MAX (3)   BYTE INITIAL (252,181,192),
       ENTA ADDRESS,ENT BASED ENTA BYTE,
```

```
     VALA ADDRESS,VAL BASED VALA BYTE,
    DIFFA ADDRESS,DIFF BASED DIFFA BYTE,
    VAL1A ADDRESS, VAL1 BASED VAL1A BYTE,
    (LCCKI,MAXI) BYTE,
    DIFF1A ADDRESS,DIFF1 BASED DIFF1A BYTE;
   /*  INPUT  CONSISTS OF A BASED VARIABLE,  OUTPUT WILL
     BE IN GLOBAL VARIABLES T, T1, T2.  AND Z, Z1, Z2,
           FOR RHO AND INVS RESPECTIVELY    */
   DECLARE XA ADDRESS, X BASED XA BYTE;

  ENTA=.ETWO;  VALA=.VTWO;  DIFFA=.DTWO;
    VAL1A=V1(I);    DIFF1A=D1(I);
   LCCKI=LOOK(I);
    MAXI=MAX(I);
   ENTRY=X; ENTRY1=X(1); ENTRY2=X(2);
   IF(LOOKI=MAXI) THEN LOOKI=0;
   ID=COMPARE(.ENTRY,ENTA+LOOKI);
        IF(ID=1) THEN GOTO EXIT1;
   /*   READ UPWARDS IN THE TABLE*/
         IF(ID>1) THEN DO;
LOOP1:   IF (LOCKI=MAXI) THEN GO TO EXIT1;
            LOOKI=LOCKI+3;
   ID=COMPARE(.ENTRY,ENTA+LOOKI);
              IF(ID=1) THEN GO TO EXIT1;
                IF(ID>1) THEN GO TO LOOP1;
           LOOKI=LOOKI-3;  GO TO EXIT2;
          END;
    /*   READ DOWNWARDS IN THE TABLE    */
LOOP2:   IF(LOOKI=0) THEN GO TO EXIT1;
          LOOKI=LOOKI-3;
    ID=COMPARE(.ENTRY,ENTA+LOOKI);
               IF(ID=1) THEN GO TO EXIT1;
                 IF(ID<1) THEN GOTO LOOP2;
                  GOTO EXIT2;

EXIT1:  T=VAL(LOOKI);  T1=VAL(LOOKI+1);T2=VAL(LOOKI+2);
    IF I<>1 THEN GO TO RET;
   Z=VAL1(LOOKI);  Z1=VAL1(LOOKI+1);   Z2=VAL1(LOOKI+2);
   GO TO RET;

   EXIT2:  TEMP=ENT(LOOKI);  TEMP1=ENT(LOOKI+1);
    TEMP2=ENT(LOOKI+2);
      /*   CHANGE THE SIGN OF TEMP    */
          TEMP2=TEMP2 XOR 80H;
     /*   ENTRY - ENT(LOOK)  */
          CALL ADD(.TEMP,.ENTRY,.TEMP);

      /*  (ENTRY - ENT(LOOK))* DIFF(LOOK)    */
      CALL MULT(.TEMP,DIFFA+LOOKI,.T);
      CALL ADD(.T,VALA+LOOKI,.T);
      /*  VAL(LOOK)+ENTRY-ENT(LOOK) *DIFF(LOOK)    */
  IF I<>1 THEN GO TO RET;
       CALL MULT(.TEMP,DIFF1A+LOOKI,.Z);
     CALL ADD(.Z,VAL1A+LOOKI,.Z);
  RET: LOCK(I)=LOOKI;
      RETURN;
 END FUNCTION;

COSSIN: PROCEDURE(XA);
   /*  THIS ROUTINE  COMPUTES  BOTH  THE  SINE  AND
       THE  COSINE  FOR  THE  GIVEN  ANGLE  XA  IN
        RADIANS .  THE  OUTPUT   VALUES  ARE
        STORED  IN  GLOBAL  VARIABLES  Z,Z1,Z2
        FOR     COSINE   T,T1,T2  FOR SINE  */
    DECLARE  (TH,TH1,TH2,THN,THN1,THN2,TMP,TMP1,TMP2)
    BYTE,
    (MPI2,MPI21,MPI22)    BYTE  INITIAL (0C9H,0FH,0C1H) ,
               XA  ADDRESS , X BASED  XA  BYTE;
   TH=X; TH1=X(1); TH2=X(2);
    DECLARE OUT LABEL;
    DECLARE I BYTE;
    DO I=1 TO 4;
```

```
        THN=TH;  THN1=TH1 ; THN2=TH2;
        CALL ADD (.TH,.MPI2,.TH);/*THETA - PI2 */
                 /*  THN IS THBAR AND TH IS THBAR - PI/2  */
        IF(ZE AND 80H) <> 0 THEN GO TO OUT;
        END;
        RETURN;
    CUT:
           TH2=ZE AND 7FH;
        CALL FUNCTION (2,.THN);
        THN=T ;  THN1=T1;  THN2=T2;
   /* THN= COS (TH )    */
        CALL FUNCTION (2,.TH);
      /*   TH= COS PI/2-TH)       */
        TH=T;    TH1=T1; TH2=T2;
        IF ((I=1)OR (I=3)) THEN DO;
               Z=THN; Z1=THN1 ;Z2=THN2;
               T=TH;  T1=TH1 ;  T2=TH2;
               IF(I=3) THEN DO;
                  Z2=Z2 XOR 80H;
                  T2=T2 XOR 80H;
                  RETURN;
                  END;
               RETURN;
               END   ;
        Z=TH;  Z1=TH1; Z2=TH2 XOR 80H;
        T=THN;  T1=THN1; T2=THN2 ;
        IF (I=4) THEN DO;
                  T2=T2 XOR 80H;
                  Z2=Z2 XOR 80H;
                  END;
        RETURN;
   END CCSSIN;

 /* DECLARATION STATEMENTS FOR THE SETDAT PROCEDURE*/
DECLARE (G,RAD,A1,AA,YT,VYK,FRACT) (3) BYTE;
DECLARE (DS2,CFORM1,CFORM2,DM1,DM2,DKG1,DKG2,VMUZ,VE,SL,FN,
        TM,DMAX) (3) BYTE;
DECLARE (ITYPE,IBOTH,J,SET) BYTE;
DECLARE (VKTS,ALT,DEG) (60) BYTE;
DECLARE (U,DEL,TEMP1X,TEMP2X,TEMP3,TEMP4,TEMP5,TEMP6,TEMP7,
        V,THETA,VXA,VYA) (3) BYTE;
 /* DECLARATION STATEMENTS FOR THE DECODE PROCEDURE*/
DECLARE IREF BYTE;
DECLARE (DTI,DS) (3) BYTE;
DECLARE (CC) (81) BYTE;
DECLARE (CT) (18) BYTE;
 /* DECLARATION STATEMENTS FOR THE TRAJ PROCEDURE */
DECLARE (CF,DM,DKG,VX,VY,TH,Y,YA,  D,DTV) (3) BYTE;
DECLARE (SWITCH,MSTG,TABLE1) BYTE;
DECLARE (TEMP1A,TEMP2A,TEMP3A,TEMP4A,TEMP5A,TEMP6A,TEMP7A)
        (3) BYTE;
 /* DECLARATION STATEMENTS FOR THE RUNGE PROCEDURE */
DECLARE (AD,Y0,VX0,VY0,RH0,AP1,AP2,AN1,AN2) (3) BYTE;
DECLARE (TEMP1B,TEMP2B,TEMP3B,TEMP4B,TEMP5B,TEMP6B) (3)BYTE;
 /* DECLARATION STATEMENTS FOR THE MAIN PROGRAM */
DECLARE (KIV,IIDNO,KIDNO,IANG,KDEG,IV,KIV,IY,KIY) BYTE;
DECLARE (NUMID) (40) BYTE;
 /* DECLARATION STATEMENTS FOR THE DERIV PROCEDURE */
DECLARE (CM,HH,CKDG) (3) BYTE;
DECLARE (IREG,BASEADDRESS) BYTE;
SETDAT: PROCEDURE;
DECLARE CONSTANTS1 (21) BYTE
     INITIAL (080H,0B2H,046H,08EH,0FAH,03BH,0B3H,033H,040H,
             0B6H,0DBH,040H,
              000H,000H,000H,0A0H,000H,0C3H,080H,000H,040H);
DECLARE CONSTANTS2 (6) BYTE
     INITIAL (000H,000H,000H,0A0H,000H,043H);
DECLARE CONSTANTS3 (2) BYTE
     INITIAL (3,1);
DECLARE PI2 (3) BYTE
     INITIAL (0C9H,00FH,043H);
 /* SET THE CONSTANTS TO THEIR SELECTED VALUES */
```

```
 IF (SET=1) THEN DO J=0 TO 2 BY 1;
     G(J)= CONSTANTS1(J);
     RAD(J)= CONSTANTS1(J+3);
     A1(J)= CONSTANTS1(J+6);
     AA(J)= CONSTANTS1(J+9);
     YT(J)= CONSTANTS1(J+12);
     VYK(J)= CONSTANTS1(J+15);
     FRACT(J)= CONSTANTS1(J+18); END; ELSE
 /* SET THE VARIABLES AS ASSIGNED IN THE DECODE PROCEDURE */
 IF (SET=2) THEN DO; DO J=0 TO 2 BY 1;
     CFCRM1(J)= CONSTANTS2(J);
     CFCRM2(J)= CONSTANTS2(J);
     DM1(J)= CONSTANTS2(J);
     DM2(J)= CONSTANTS2(J);
     DKG1(J)= CONSTANTS2(J);
     DKG2(J)= CONSTANTS2(J);
     VMUZ(J)= CONSTANTS2(J);
     VE(J)= CONSTANTS2(J);
     SL(J)= CONSTANTS2(J);
     FN(J)= CONSTANTS2(J);
     TM(J)= CONSTANTS2(J);
     DMAX(J)= CONSTANTS2(J+3); END;
     ITYPE= CONSTANTS3(0);
     IBOTH= CONSTANTS3(1); END; ELSE
 IF (SET=3) THEN DO;
DECLARE CONVERT (3) BYTE
     INITIAL (0D8H,009H,041H);
 /* CONVERT KNOTS TO FEET PER SECOND  U=VKTS(IV)*1.6878 */
     CALL MULT(.CONVERT,.VKTS,.U);
 /* APPROXIMATE THE ARCTAN FUNCTION */
     CALL DIV(.VE,.U,.DEL);
 /* DETERMINE THE VELOCITY  V=U+1/2*(VE**2)/U
           APPROX TO SQRT(U**2+VE**2) */
     CALL MULT(.VE,.VE,.TEMP1X);
     CALL DIV(.TEMP1X,.U,.TEMP2X);
     CALL MULT(.TEMP2X,.FRACT,.TEMP3);
     CALL ADD(.TEMP3,.U,.V);
 /* CALCULATE THE DIVE ANGLE      THETA= DEG(IANG)*RAD      */
     CALL MULT(.RAD,.DEG,.THETA);
 /* IF THE DIVE ANGLE IS NEGATIVE, ADD IT TO 2PI */
       IF (THETA(2) AND 80H) <> 0 THEN DO;
        CALL ADD(.THETA,.PI2,.THETA);
      END;
 /* CALCULATE THE VELOCITY IN BOTH THE X AND Y DIRECTIONS */
 /* VXA= (V+VMUZ)* COS(THETA-DEL) */
 /* VYA= (V+VMUZ)* SIN(THETA-DEL) */
     CALL SUB(.THETA,.DEL,.TEMP4);
     CALL COSSIN(.TEMP4);
      TEMP5=Z; TEMP5(1)=Z1; TEMP5(2)=Z2;
      TEMP7=T; TEMP7(1)= T1; TEMP7(2)=T2;
     CALL ADD(.V,.VMUZ,.TEMP6);
     CALL MULT(.TEMP6,.TEMP5,.VXA);
     CALL MULT(.TEMP6,.TEMP7,.VYA); END; RETURN;
END SETCAT;
DECODE: PROCEDURE;
DECLARE (START,ONE,TWO,THREE,FOUR) LABEL;
 /* DECLARE THE STARTING POSITION OF THE VARIOUS
                WEAPON COEFFICIENTS */
 DECLARE IDVEC (28) ADDRESS
  INITIAL(0,13,26,39,52,65,78,91,104,117,130,143,156,169,
          182,195,208,221,
          254,287,320,353,386,419,452,485,518,551);
 /* DECLARE THE MUZZLE VELOCITY AND THRUST VECTOR */
DECLARE VMFN (6) BYTE
  INITIAL(0CEH,040H,04CH,0DAH,040H,04BH);
 /* DECLARE THE CC MATRIX OF DRAG COEFFICIENTS */
DECLARE CCVALUE (81) BYTE
  INITIAL(0CEH,02AH,037H,000H,000H,000H,000H,000H,000H,0BFH,
         0A0H,03CH,
         0E0H,080H,0BDH,088H,04AH,03DH,0EEH,058H,0BDH,0DEH,
         0DEH,03EH,
         0C8H,007H,0BDH,0E2H,03EH,042H,0D6H,042H,0C2H,0B7H,
```

```
            0D9H,042H,
            0B4H,02FH,044H,0DBH,054H,0C5H,0ADH,0D8H,045H,0BEH,
            055H,0C5H,
            0B1H,00BH,046H,0E7H,0FCH,0C4H,0D5H,03AH,03DH,0EBH,
            0E0H,0BEH,
            0ABH,0AAH,03EH,0C6H,0B1H,0BEH,0CDH,08EH,03FH,0A8H,
            090H,0BEH,
            096H,028H,03DH,0A6H,085H,0BBH,0A0H,05AH,038H);
 /* DECLARE THE MACH CUT MATRIX CT */
DECLARE CTVALUE (18) BYTE
   INITIAL(0D5H,081H,040H,0FAH,01CH,040H,09FH,03BH,040H,0E2H,
            08FH,040H,
            084H,018H,041H,0A6H,066H,041H);
 /* DECLARE THE WPNCODE MATRIX CONTAINING THE VARIABLE FOR
                                           EACH WEAPON */
DECLARE WPNCODE (585) BYTE
 /* WEAPCN CONSTANTS FOR THE MK 43 UNRETARDED */
  INITIAL(4,000H,000H,000H,0A7H,027H,038H,0A0H,000H,043H,
            0C0H,000H,042H,
 /* WEAPCN CONSTANTS FOR THE MK 57 UNRETARDED */
            4,000H,000H,000H,0CEH,06BH,039H,0A0H,000H,043H,
            0C0H,000H,042H,
 /* WEAPCN CONSTANTS FOR THE MK 61 UNRETARDED */
            4,000H,000H,0C0H,083H,066H,039H,0A0H,000H,043H,
            0C0H,000H,042H,
 /* WEAPCN CONSTANTS FOR THE MK 116 WETEYE */
            2,080H,090H,039H,0B4H,07CH,038H,0C0H,000H,042H,
            08CH,000H,042H,
 /* WEAPCN CONSTANTS FOR THE MK 76 WITH LUG */
            2,080H,00CH,039H,0D0H,090H,039H,0C0H,000H,042H,
            0CCH,000H,042H,
 /* WEAPCN CONSTANTS FOR THE MK 77 FIREBOMB */
            4,000H,000H,0C0H,0AEH,036H,038H,080H,000H,042H,
            080H,000H,041H,
 /* WEAPCN CONSTANTS FOR THE MK 81 */
            1,0A4H,081H,042H,000H,000H,000H,0A0H,000H,043H,
            0CCH,000H,042H,
 /* WEAPCN CONSTANTS FOR THE MK 81 SNAKEYE UNRETARDED */
            4,000H,000H,000H,0A0H,005H,03AH,0CCH,00CH,042H,
            08CH,000H,042H,
 /* WEAPCN CONSTANTS FOR THE MK 82 MECH FUZE */
            1,084H,018H,042H,000H,000H,000H,0A0H,000H,043H,
            0C0H,000H,042H,
 /* WEAPCN CONSTANTS FOR THE MK 82 ELECT FUZE */
            1,0BFH,021H,041H,000H,000H,0C0H,0A0H,000H,043H,
            0C0H,000H,042H,
 /* WEAPCN CONSTANTS FOR THE MK 83 MECH FUZE */
            1,0ABH,0EAH,041H,000H,000H,000H,0A0H,000H,043H,
            0C0H,000H,042H,
 /* WEAPCN CONSTANTS FOR THE MK 83 ELEC FUZE */
            1,09AH,0E1H,041H,000H,000H,000H,0A0H,000H,043H,
            0C0H,000H,042H,
 /* WEAPON CONSTANTS FOR THE MK 84 */
            1,080H,000H,041H,000H,000H,000H,0A0H,000H,043H,
            0C0H,000H,042H,
 /* WEAPCN CONSTANTS FOR THE MK 117 AL */
            1,0C7H,0AEH,042H,0A0H,04DH,0B7H,0A0H,000H,043H,
            0CCH,000H,042H,
 /* WEAPCN CONSTANTS FOR THE MK 86 WET SAND FILLEC */
            1,0DFH,0D2H,042H,000H,000H,000H,0C0H,00CH,042H,
            08CH,000H,042H,
 /* WEAPON CONSTANTS FOR THE MK 88 WET SAND FILLED */
            1,0CDH,070H,041H,000H,000H,000H,0A0H,000H,043H,
            0CCH,000H,042H,
 /* WEAPON CONSTANTS FOR THE MK 82 SNAKEYE UNRETARDED */
            4,000H,000H,000H,0F0H,028H,039H,0C0H,000H,042H,
            08CH,000H,041H,
 /* WEAPCN CONSTANTS FOR THE MK 82 SNAKEYE RETARDED */
            1,000H,000H,000H,0F0H,028H,039H,000H,000H,000H,
            08CH,000H,042H,
          1,2,000H,000H,000H,08AH,067H,03BH,0C2H,08FH,03FH,
            0AFH,0C7H,03EH,
```

```
            0A9H,065H,040H,08DH,008H,0B5H,
/* WEAFCN CCNSTANTS FOR THE SADEYE T1= 4.0 */
          1,084H,0D3H,042H,000H,000H,000H,000H,000H,000H,
          0CCH,000H,041H,
        1,2,0C0H,000H,000H,0E3H,005H,03EH,000H,000H,000H,
          000H,000H,000H,
            088H,03BH,043H,000H,000H,000H,
/* WEAPCN CONSTANTS FOR THE ROCKEYE II T1= 4.0 */
          1,093H,006H,042H,085H,0F0H,03AH,000H,00CH,000H,
          08CH,000H,042H,
        1,2,0A3H,0D7H,03FH,0B6H,073H,03AH,0D1H,0EBH,03FH,
          0ACH,0E7H,03EH,
            081H,0FBH,043H,000H,000H,000H,
/* WEAFCN CONSTANTS FOR THE CBU T1= 4.0 */
          1,08FH,062H,042H,000H,000H,000H,000H,000H,000H,
          0CFH,05CH,041H,
        1,2,000H,000H,000H,0F1H,041H,03DH,000H,00CH,000H,
          00CH,000H,000H,
            080H,000H,043H,000H,000H,000H,
/* WEAPCN CONSTANTS FOR THE MK 81 SNAKEYE RETARCED */
          1,000H,000H,000H,0A0H,005H,03AH,000H,00CH,000H,
          0CFH,09DH,041H,
        1,2,000H,000H,000H,0BCH,0EDH,03BH,0C2H,08FH,03FH,
          0EEH,075H,03EH,
            0ADH,0D2H,040H,09EH,0DBH,0B5H,
/* WEAPCN CCNSTANTS FOR THE 20 MM GUN */
          3,0BFH,0C5H,042H,0F5H,0A1H,0BAH,0C0H,00CH,041H,
          080H,000H,040H,
        3,1,000H,000H,000H,000H,000H,000H,000H,00CH,000H,
          00CH,000H,000H,
            000H,000H,000H,00CH,00CH,000H,
/* WEAFCN CONSTANTS FOR THE 5 INCH ROCKETS */
          3,0D1H,0EBH,040H,000H,000H,000H,000H,00CH,000H,
          08CH,000H,041H,
        2,1,000H,000H,000H,080H,000H,041H,000H,00CH,000H,
          000H,000H,000H,
            0B6H,014H,041H,000H,000H,000H,
/* WEAPCN CONSTANTS FOR THE MK 43 RETARDED 0.4 SEC DELAY */
          4,000H,000H,000H,000H,000H,000H,000H,00CH,000H,
          09EH,0B8H,03FH,
        0,1,000H,000H,000H,000H,000H,000H,000H,000H,000H,
          0BDH,070H,041H,
            0FAH,0E1H,040H,000H,000H,000H,
/* WEAPCN CONSTANTS FOR THE MK 57 RETARDED 0.8 SEC DELAY */
          4,000H,000H,000H,000H,000H,000H,000H,000H,000H,
          0E1H,047H,03EH,
        0,1,000H,000H,000H,000H,000H,000H,000H,000H,000H,
          08CH,000H,042H,
            0E3H,0D7H,040H,000H,000H,000H,
/* WEAPCN CONSTANTS FOR THE MK 61 RETARDED 0.6 SEC DELAY */
          4,000H,000H,000H,000H,000H,000H,000H,00CH,000H,
          0CCH,0CCH,03DH,
        0,1,000H,000H,000H,000H,000H,000H,000H,000H,000H,
          0ACH,0CCH,042H,
            0E3H,0D7H,040H,000H,000H,000H,
/* WEAFCN CONSTANTS FOR THE MK 106 MOD 2 */
          2,09BH,008H,03EH,00CH,000H,000H,000H,00CH,000H,
          0CCH,0CCH,040H,
        2,1,000H,000H,000H,09BH,008H,03EH,000H,000H,000H,
          000H,000H,000H,
            080H,000H,040H,000H,000H,000H);
/* ASSIGN THE REFERENCE VALUE FROM THE WPNCODE */
  IREF= WPNCODE(IDVEC(IDNO-1));
/* ASSIGN THE VARIABLES THEIR RESPECTIVE VALUES FROM THE
                WPNCODE */
   DC J=C TO 2 BY 1;
    CFCRM1(J)= WPNCODE(IDVEC(IDNO-1)+1+J);
    DKG1(J)= WPNCODE(IDVEC(IDNO-1)+4+J);
    CMAX(J)= WPNCODE(IDVEC(IDNO-1)+7+J);
    DTI(J)= WPNCODE(IDVEC(IDNO-1)+10+J);
   ENC;
/* DECIDE IF SINGLE DRAG WEAPON AND THEN BRANCH
```

```
                 ACCORDINGLY  */
IF (IDNO <= 17) THEN GO TO START; ELSE DO;
 /* ASSIGN THE SECOND PORTION OF THE VARIABLES FOR DUAL
                 STAGE WEAPONS */
        ITYPE= WPNCODE(IDVEC(IDNO-1)+13);
        IBOTH= WPNCODE(IDVEC(IDNO-1)+14);
       DO J=0 TO 2 BY 1;
        DM1(J)= WPNCODE(IDVEC(IDNO-1)+15+J);
        CFORM2(J)= WPNCODE(IDVEC(IDNO-1)+18+J);
        DM2(J)= WPNCODE(IDVEC(IDNO-1)+21+J);
        DKG2(J)= WPNCODE(IDVEC(IDNO-1)+24+J);
        DS(J)= WPNCODE(IDVEC(IDNO-1)+27+J);
        SL(J)= WPNCODE(IDVEC(IDNO-1)+30+J);
       END; END;
 /* IF IDNO=23 ASSIGN THE MUZZLE VELOCITY TO VMUZ */
IF (IDNO = 23) THEN DO J=0 TO 2 BY 1; VMUZ(J)= VMFN(J); END;
 /* IF THE IDNO=24 ASSIGN THE THRUST TO FN */
IF (IDNO = 24) THEN DO J=0 TO 2 BY 1; FN(J)= VMFN(J+3); END;
 /* BRANCH TO THE APPROPRIATE SECTION OF THE CC MATRIX */
START: IF (IREF = 1) THEN GO TO ONE;
         IF (IREF = 2) THEN GO TO TWO;
         IF (IREF = 3) THEN GO TO THREE;
         IF (IREF = 4) THEN GO TO FOUR;
ONE: DO J=0 TO 53 BY 1; CC(J)= CCVALUE(J); END;
     DO J=0 TO 11 BY 1; CT(J)= CTVALUE(J); END;
     GO TO FOUR;
TWO: DO J=0 TO 26 BY 1; CC(J)= CCVALUE(J+27); END;
     DO J=0 TO 5 BY 1; CT(J)= CTVALUE(J+6); END;
     GO TO FOUR;
THREE: DO J=0 TO 26 BY 1; CC(J)= CCVALUE(J+54); END;
         DO J=0 TO 5 BY 1; CT(J)= CTVALUE(J+12); END;
FOUR: RETURN;
END DECODE;
INPUT: PROCEDURE;
     DECLARE J BYTE;
     DO J=0 TO 2;
     VKTS(J)=INDAT(J);
     DEG(J)=INDAT(3+J);
     ALT(J)=INDAT(6+J);
     END;
END INPUT;
DERIV: PROCEDURE;
DECLARE (TEMP1C,TEMP2C,TEMP3C,TEMP4C,TEMP5C) (3) BYTE;
DECLARE CONSTANTS6 (6) BYTE
      INITIAL (0EAH,0BBH,036H,0E0H,006H,024H);
DECLARE (EIGHT,NINE,TEN) LABEL;
                              V=SQRT(VX*VX+VY*VY) */
 /* COMPUTE THE VELOCITY OF THE WEAPON
     CALL MULT(.VX,.VX,.TEMP1C);
     CALL MULT(.VY,.VY,.TEMP2C);
     CALL ADD(.TEMP1C,.TEMP2C,.TEMP3C);
     CALL SQRT(.TEMP3C,.V);
 /* COMPUTE THE MACH OF THE WEAPON
               CM=V*(8.955E-04+3.26E-09*Y)+DM */
     DO J=0 TO 2 BY 1;
      TEMP1C(J)= CONSTANTS6(J);
      TEMP2C(J)= CONSTANTS6(J+3);
     END;
     CALL MULT(.Y,.TEMP2C,.TEMP3C);
     CALL ADD(.TEMP3C,.TEMP1C,.TEMP4C);
     CALL MULT(.V,.TEMP4C,.TEMP5C);
     CALL ADD(.TEMP5C,.DM,.CM);
 /* DETERMINE THE REGION OF THE DRAG CURVE WHICH IS
         APPLICABLE */
 IF (2 = COMPARE(.CM,.CT(MSTG))) THEN GO TO EIGHT;
     IREG=0; GO TO TEN;
EIGHT:  IF (2 = COMPARE(.CM,.CT(MSTG+3))) THEN GO TO NINE;
     IREG= 9; GO TO TEN;
NINE:  IREG= 18;
 /* DO THE INTERMEDIATE BALLISTIC CALCULATIONS */
 /* CKDG=DKG+CF*(CC(IREG,1,MSTG)+(CC(IREG,2,MSTG)+
              CC(IREG,T,MSTG)*CM)*CM) */
```

```
TEN:   BASEADDRESS= TABLE1+IREG;
     CALL MULT(.CM,.CC(BASEADDRESS+6),.TEMP1C);
     CALL ADD(.TEMP1C,.CC(BASEADDRESS+3),.TEMP2C);
     CALL MULT(.CM,.TEMP2C,.TEMP3C);
     CALL ADD(.TEMP3C,.CC(BASEADDRESS),.TEMP4C);
     CALL MULT(.CF,.TEMP4C,.TEMP5C);
     CALL ADD(.TEMP5C,.DKG,.CKDG);
 /* HH= TH/V-RHO*CKDG*V   */
     CALL MULT(.V,.CKDG,.TEMP1C);
     CALL MULT(.TEMP1C,.RHO,.TEMP2C);
     CALL DIV(.TH,.V,.TEMP3C);
     CALL SUB(.TEMP3C,.TEMP2C,.HH);
 /* AN2= HH*VX   */
     CALL MULT(.HH,.VX,.AN2);
 /* AP2= HH*VY-G */
     CALL MULT(.HH,.VY,.TEMP4C);
     CALL SUB(.TEMP4C,.G,.AP2);
RETURN; END DERIV;
RUNGE: PROCEDURE;
DECLARE CONSTANTS5 (9) BYTE
  INITIAL(09BH,082H,038H,093H,0A6H,029H,0BDH,00BH,018H);
 /* CALCULATE THE AD VALUE       AD= A*D      */
     CALL MULT(.A1,.D,.AD);
 /* ASSIGN THE VARIABLES THEIR INITIAL VALUES       */
     DO J=0 TO 2 BY 1;
      YC(J)= Y(J);
      VXC(J)= VX(J);
      VYC(J)= VY(J);
     END;
 /* CALCULATE THE AIR DENSITY
        RHO=2.37E-03-Y*(6.87E-08-Y*6.71E-13) */
     DO J=0 TO 2 BY 1;
      TEMP1B(J)= CONSTANTS5(J);
      TEMP2B(J)= CONSTANTS5(J+3);
      TEMP3B(J)= CONSTANTS5(J+6);
     END;
     CALL MULT(.Y,.TEMP3B,.TEMP4B);
     CALL SUB(.TEMP2B,.TEMP4B,.TEMP5B);
     CALL MULT(.Y,.TEMP5B,.TEMP6B);
     CALL SUB(.TEMP1B,.TEMP6B,.RHO);
 /* MAKE THE FIRST CALL TO THE DERIV PROCEDURE */
CALL DERIV;
 /* UPDATE THE POSITIONS AND THE VELOCITIES       */
 /* Y= YC+AD*VY */
     CALL MULT(.AD,.VY,.TEMP1B);
     CALL ADD(.TEMP1B,.YO,.Y);
     DO J=0 TO 2 BY 1;
      AP1(J)= AP2(J);
      AN1(J)= AN2(J);
     END;
 /* VX= VXO+AD*AN1 */
     CALL MULT(.AN1,.AD,.TEMP1B);
     CALL ADD(.VXO,.TEMP1B,.VX);
 /* VY= VYC+AD*AP1 */
     CALL MULT(.AP1,.AD,.TEMP2B);
     CALL ADD(.VYO,.TEMP2B,.VY);
 /* CALCULATE THE AIR DENSITY
        RHO=2.37E-03-Y*(6.87E-08-Y*6.71E-13) */
     DO J=0 TO 2 BY 1;
      TEMP1B(J)= CONSTANTS5(J);
      TEMP2B(J)= CONSTANTS5(J+3);
      TEMP3B(J)= CONSTANTS5(J+6);
     END;
     CALL MULT(.Y,.TEMP3B,.TEMP4B);
     CALL SUB(.TEMP2B,.TEMP4B,.TEMP5B);
     CALL MULT(.Y,.TEMP5B,.TEMP6B);
     CALL SUB(.TEMP1B,.TEMP6B,.RHO);
 /* MAKE THE SECOND CALL TO THE DERIV PROCEDURE      */
CALL DERIV;
 /* COMPUTE THE TIME, POSITION AND VELOCITIES      */
 /* T= T+D   */
     CALL ADD(.TM,.D,.TM);
```

```
 /* X= X+D*(VXO+AA*(VX-VXO))  */
     CALL SUB(.VX,.VXO,.TEMP2B);
     CALL MULT(.AA,.TEMP2B,.TEMP3B);
     CALL ADD(.VXO,.TEMP3B,.TEMP4B);
     CALL MULT(.TEMP4B,.D,.TEMP5B);
     CALL ADD(.X,.TEMP5B,.X);
 /* Y= YO+D*(VYO+AA*(VY-VYO))  */
     CALL SUB(.VY,.VYO,.TEMP2B);
     CALL MULT(.AA,.TEMP2B,.TEMP3B);
     CALL ADD(.VYO,.TEMP3B,.TEMP4B);
     CALL MULT(.TEMP4B,.D,.TEMP5B);
     CALL ADD(.YO,.TEMP5B,.Y);
 /* VX= VXO+D*(AN1+AA*(AN2-AN1))  */
     CALL SUB(.AN2,.AN1,.TEMP2B);
     CALL MULT(.TEMP2B,.AA,.TEMP3B);
     CALL ADD(.TEMP3B,.AN1,.TEMP4B);
     CALL MULT(.D,.TEMP4B,.TEMP5B);
     CALL ADD(.VXO,.TEMP5B,.VX);
 /* VY= VYO+D*(AP1+AA*(AP2-AP1))  */
     CALL SUB(.AP2,.AP1,.TEMP2B);
     CALL MULT(.TEMP2B,.AA,.TEMP3B);
     CALL ADD(.AP1,.TEMP3B,.TEMP4B);
     CALL MULT(.D,.TEMP4B,.TEMP5B);
     CALL ADD(.VYO,.TEMP5B,.VY);
RETURN;
END RUNGE;
TRAJ: PROCEDURE;
DECLARE (FIVE,SIX,SEVEN) LABEL;
DECLARE CONSTANTS4 (4) BYTE
     INITIAL (000H,000H,000H,0);
 /* INITIALIZE THE VARIABLES FOR THE TRAJECTORY PROCEDURE */
     MSTG= CONSTANTS4(3);
     TABLE1= CONSTANTS4(3);
     DO J=0 TO 2 BY 1;
     CF(J)= CFORM1(J);
     DM(J)= DM1(J);
     DKG(J)= DKG1(J);
     VX(J)= VXA(J);
     VY(J)= VYA(J);
     TH(J)= FN(J);
     Y(J)= ALT(J);
     YA(J)= Y(J);
     X(J)= CONSTANTS4(J);
     TM(J)= CONSTANTS4(J);
     END;
 /* DETERMINE THE TYPE OF DRAG */
 IF (ITYPE=3) THEN GO TO FIVE;
 /* CALCULATE THE STEP SIZE  D=DS+SL*U  */
     CALL MULT(.SL,.U,.TEMP1A);
     CALL ADD(.TEMP1A,.DS,.D);
      GO TO SIX;
 /* SET THE STEP SIZE TO THE MAX ALLOWED  D= DMAX  */
 FIVE: DO J=0 TO 2 BY 1; D(J)= DMAX(J); END;
 /* CALL THE RUNGE PROCEDURE FOR THE INTEGRATION */
  SIX: CALL RUNGE;
 /* CALCULATE THE DTV VALUE  DTV= 1/G*(VY+SQRT(VY**2+
       2.*G*Y)) */
     CALL MULT(.G,.Y,.TEMP2A);
     TEMP3A= TEMP2A; TEMP3A(1)= TEMP2A(1);
     TEMP3A(2)=TEMP2A(2)+1;
     CALL MULT(.VY,.VY,.TEMP4A);
     CALL ADD(.TEMP4A,.TEMP3A,.TEMP5A);
     CALL SQRT(.TEMP5A,.TEMP6A);
     CALL ADD(.TEMP6A,.VY,.TEMP7A);
     CALL DIV(.TEMP7A,.G,.DTV);
     DO J=0 TO 2 BY 1; D(J)= DTI(J); END;
IF (IDNC <= 17) THEN GO TO SEVEN;
IF (IDNC = 23) THEN GO TO SEVEN;
 /* SET THE SECOND STAGE DRAG PARAMETERS */
     MSTG=6; TABLE1=27;
     IF (ITYPE=2) THEN DO; MSTG=0; TABLE1= 0; END;
     DO J=0 TO 2 BY 1;
```

```
      DKG(J)= DKG2(J);
      DM(J)= DM2(J);
      CF(J)= CFORM2(J);
      TH(J)= CONSTANTS4(J); END;
 /* TEST THE STEP SIZE VERSUS THE VACUMN FALL STEP SIZE  */
 SEVEN: IF (0<>COMPARE(.DTV,.D)) THEN GO TO SIX;
 /* SET THE STEP SIZE TO THE VACUMN VALUE */
     DO J=0 TO 2 BY 1;
      D(J)= DTV(J);
     END;
 /* SET THE DRAG PARAMETERS FOR THE FINAL INTEGRATION
          STEP */
     MSTG= 6; TABLE1= 27;
     IF (ITYPE=2) THEN DO; MSTG= 0; TABLE1=0; END;
     DO J=0 TO 2 BY 1;
      DKG(J)= DKG2(J);
      DM(J)= DM2(J);
      TH(J)= CONSTANTS4(J);
      CF(J)= CFORM2(J);
     END;
 /* CALL RUNGE FOR THE FINAL INTEGRATION  */
CALL RUNGE;
 /* CALCULATE THE DTV VALUE  DTV= 1/G*(VY+SQRT(VY**2+
         2.*G*Y))  */
     CALL MULT(.G,.Y,.TEMP2A);
     TEMP3A= TEMP2A; TEMP3A(1)= TEMP2A(1);
             TEMP3A(2)=TEMP2A(2)+1;
     CALL MULT(.VY,.VY,.TEMP4A);
     CALL ADD(.TEMP4A,.TEMP3A,.TEMP5A);
     CALL SQRT(.TEMP5A,.TEMP6A);
     CALL ADD(.VY,.TEMP6A,.TEMP7A);
     CALL DIV(.TEMP7A,.G,.DTV);
 /* UP DATE THE TIME OF FALL OF THE WEAPON  TM= TM+DTV  */
     CALL ADD(.DTV,.TM,.TM);
 /* UP-DATE THE DOWN RANGE TRAVEL OF THE WEAPON
           X=X+DTV*VX  */
     CALL MULT(.DTV,.VX,.TEMP2A);
     CALL ADD(.X,.TEMP2A,.X);
 /*  SET THE SWITCH FOR THE PRINT PROCEDURE  */
     SWITCH= 0;
RETURN;
END TRAJ;
     DECLARE CTE(3) BYTE;
     CTE=0C0H;
     CTE(1)=0H;
     CTE(2)=42H;
         SET=1;
         CALL SETDAT;
     CALL INPUT;
         SET=2;
         CALL SETDAT;
     CALL DECODE;
         SET=3;
     CALL SETDAT;
         CALL TRAJ;
     CALL DIV(.X,.CTE,.X);
 END;
             OUTPUT(6)=0FFH; /* INTERRUPT */
             OUTPUT(6)=0H;  /* CLEARS INTERRUPT */
             DO WHILE INPUT(7)<>INTER;  END;
             DO I=0 TO 2;
               DO WHILE INPUT(7) <> I+1;  END;
               OUTPUT(7)=X(I);
             END;
             COUNT=COUNT+1;
             CALL TIME(5);
             /* WAITS FOR REPLY FROM EXEC.  */
             IF INPUT(7)=FROM$BUFFER THEN DO;
             OUTPUT(7)=FROM$BUFFER;
             /*  TELLS EXEC TO SEND DATA  */
             GO TO LOOP;
             END;
```

```
        RETURN;
        END SERV$TWO;


DO FOREVER;
COUNT=0;
OUTPUT(6)=0;  /* CLEARS INTERRUPT */
OUTPUT(7)=INIT;
/* THIS MEANS: BAL IS READY  */
DO WHILE INPUT(7) <> INIT;  END;
/* WAITS FOR THE IDNO */
CALL SERV$ONE;
DO WHILE COUNT <> 5;
/* COUNT IS CONTROLLED BY SERV$TWO PROCEDURE  */
OUTPUT(7)=DAT;
DO WHILE INPUT(7) <> DAT;  END;
/* WAITS FOR DATA  */
CALL SERV$TWO;
END;  /* WHILE COUNT  */
END;  /* FOREVER  */
EOF
```

LIST OF REFERENCES

1. Jupin,H.A., The Ballistics Processor of a Multiple Processor Airborne Tactical System, MS Thesis, NPS, June 1975
2. Mc Cracken,W.L., Design Study of an Avionics Navigation Microcomputer, AE Thesis, NPS, June 1974
3. Conte, S. D. and de Boor, C., Elementary Numerical Analysis: An Algorithmic Approach, p. 191-273, Mc Graw-Hill, 1972
4. Kildall,G.A., CP/M: A Disk Control Program for Microcomputer System Development, June 1975
5. Intel Corporation, INTELLEC/8 MOD 80 Reference Manual, 1974
6. Grumman Aerospace Corporation, Navy Model A-6E Aircraft, Integrated Weapon System Theory, NAVAIR 01-85ADF-2-10.1, September, 1971

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Group Chairman, Code 72 Kr Naval Postgraduate School Computer Science Group Monterey, California 93940	1
4.	Ass. Professor Uno R. Kodres, Code 72 Kr Naval Postgraduate School Computer Science Group Monterey, California 93940	1
5.	Lcdr Tuxaua P. B. de Linhares Departament of the Navy Rio de Janeiro, Brasil	1

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA 93940

6 AUG 76	24137
8 NOV 76	24941
[illegible]	[illegible]
12 MAR 78	24037
24 JAN 79	~~25232~~
13 SEP 79	26440
29 OCT 80	26147

Thesis 163608
L6634 Linhares
c.1 Distributed micro-computer airborne tactical system.

6 AUG 76	24137
8 NOV 76	24941
[illegible]	[illegible]
12 MAR 78	24037
24 JAN 79	~~25232~~
13 SEP 79	26440
29 OCT 80	26147

Thesis 163608
L6634 Linhares
c.1 Distributed micro-computer airborne tactical system.

thesL6634
Distributed microcomputer airborne tacti
3 2768 002 11801 0
DUDLEY KNOX LIBRARY